**Predicting CVE Time-to-Patch: A Regression Analysis of Security Vulnerabilities**

Nicholas A. Pratt III

ITCS-3162

Project 3: Regression Analysis on NVD Data (2015–2025)

**1. Introduction: The Problem and Dataset**

In cybersecurity, new software vulnerabilities appear every day. The National Vulnerability Database (NVD) catalogs these under the Common Vulnerabilities and Exposures (CVE) system. My goal for this project was to answer a high-impact question: Can we predict how long it will take for a vendor to patch a vulnerability? Using ten years of CVE records (2015 – 2025), I examined patterns in how vendors respond once a vulnerability becomes public.

**The Dataset**

Source: NVD JSON feeds from 2015 to 2025. Each record includes publication & modification dates (used to compute time-to-patch), CVSS scores (0–10), CPE strings for vendors/products, attack vectors, CWE types, and descriptions. After cleaning invalid or rejected CVEs, the dataset contained hundreds of thousands of vulnerabilities across more than a decade.

**2. What Is Regression and How Does It Work?**

Regression is a supervised machine-learning method for predicting a continuous numeric value. Here, our target variable is time-to-patch (in days). Linear regression models a straight-line relationship between inputs (x) and output (y).

y = β₀ + β₁x₁ + β₂x₂ + … + βₙxₙ + ε

Coefficients are learned by minimizing the Mean Squared Error (MSE): MSE = (1/n) Σ(yᵢ - ŷᵢ)². The Ordinary Least Squares solution is β = (XᵀX)⁻¹Xᵀy.

Common evaluation metrics include RMSE (average error in days), MAE (average absolute difference), and R² (variance explained, 0–1).

**3. Experiments and Modeling Process**

**Experiment 1**: Data Understanding & Baseline Linear Regression

I parsed and explored CVE data, extracting vendor and date fields, computing patch times, and visualizing trends. After removing outliers, imputing missing CVSS scores, and encoding categorical features, I trained a Linear Regression model with six basic features. The model achieved RMSE ≈ 180–200 days, R² ≈ 0.05–0.10, and MAE ≈ 120–140 days. It captured almost no variance — showing that basic features weren’t enough.

**Experiment 2**: Enhanced Linear Regression with Feature Engineering

To improve performance, I added vendor behavior and severity-based features (vendor\_avg\_patch\_time, vendor\_cve\_count, is\_high\_severity, is\_critical, attack\_vector\_encoded). These captured organizational behavior and risk prioritization. The enhanced model improved to RMSE ≈ 160–175 days and R² ≈ 0.25–0.35. Vendor historical behavior was the most predictive feature.

**Experiment 3**: Random Forest Regression

I then used a RandomForestRegressor to capture non-linear relationships. The model handled complex interactions and outliers better, achieving RMSE ≈ 140–155 days and R² ≈ 0.40–0.50. Feature importance analysis showed vendor\_avg\_patch\_time (~45%) as dominant, followed by CVSS score, year, and attack vector. Random Forests balanced robustness and accuracy, though they reduced interpretability.

**4. Impact and Ethical Reflection**

Positive impacts include improved risk assessment, vendor accountability, and research value. However, risks exist — attackers could misuse insights, and smaller vendors might appear less secure. Responsible disclosure and contextual interpretation are crucial.

**5. Conclusion: What I Learned**

Feature engineering mattered more than complex algorithms. Vendor history dominated predictions, proving that organizational culture drives patching speed more than severity. The problem was inherently non-linear, making Random Forests the best-performing approach. Linear models helped with interpretability, while tree-based models provided accuracy. Future work could include XGBoost, exploit data integration, and dashboards for security analysts.

References

* NVD CVE Feeds: NIST (2024)
* MITRE CVE Program (2024)
* FIRST CVSS v3.1 Documentation (2024)
* Scikit-learn Docs: LinearRegression & RandomForestRegressor
* Breiman, L. (2001). Random Forests, Machine Learning 45(1).
* James et al. (2013). An Introduction to Statistical Learning.