Project 1: Prison Management System Modernization

Name: Colin Cooper

Assessment Report

1.1 Root Cause Analysis

Overview:

Direct feedback from the client and measurements of the performance of the system indicate several symptoms of project failure. These include major cost overruns against the plan for the first 4 years of the project. This is in spite of an attempt to rein in costs in Year 4 by reducing the project team size by 33% (from 42 team members to 28). The client is reporting that the benefits envisioned at the outset of the project are not being achieved and in some cases leading to operational cost increases instead of the anticipated reductions that had been expected had the system been working as expected. The client has also reported significant problems with the quality of the system including latency issues, missing information, and concerns about the usability of the system overall.

A disconnection from the requirements and issues that the client faces, as well as adopting a lift-and-shift approach to applying an AI solution that may have been appropriate in a different context has led the client to feel that the project team does not understand their real problems, and is trying to apply an inappropriate solution that doesn't properly solve those problems. This is a conclusion that is difficult to dispute, and has resulted in the project failure to meet its schedule, budget and objectives.

Cost Overruns:

The costs of the project have exceeded the budget for the first 4 years. At a budgeted average run-rate of $70m per year, the project would have anticipated total costs at the end of Year 4 of $280m. As $350m has been spent to date, the project is already 32% over budget.

While the project is currently reported as being $20m over budget, this is an over-optimistic estimate as the project has only completed 4 years of its anticipated 5-year schedule. After 4 years and using a flat run-rate, the project is actually $70m over budget and on-track to complete at $60m over the original baseline budget. (See Appendix A-1)

- $35m of the total cost overrun has been identified as coming from AI development costs and cloud compute costs.
- The loss of two devops engineers reduced the team's capacity for maintaining the system's AWS infrastructure, leading to higher latency and ultimately higher costs as poor monitoring and identification of AWS service efficiencies led to higher than needed cloud service costs. Inefficient use of expensive AWS GPUs for non-intensive tasks which cheaper CPU compute costs would have been sufficient and far less costly.
- Adoption of AI training algorithms that are inappropriate to the prison management environment and more appropriate for ultra high-volume data environments such as a bank. While these algorithms have the potential to improve the accuracy of the produced models, they come at significantly higher training and infrastructure cost while much cheaper ways of improving the accuracy of the models have been overlooked.

The root cause of the remaining $35m of cost over-run comes from several other sources, including:

- Increased cloud (AWS) storage costs for example through excessive storing of data in more expensive S3 storage classes that are designed for use-cases that require availability and resiliency features in excess of what is needed for training data.
- Increased cloud (AWS) network traffic ingress and egress costs for example through excessive movement of data in and out of the AI development, training, testing and production environments.

Schedule Overruns:
In an attempt to rein in some of the project costs, the project sponsors decided to reduce the project headcount from 42 to 28 in Year 4. Some groups of project staff were particularly impacted including Data Engineers (-50% capacity); Quality Control Analysts (-80% capacity) and cloud security specialists (-67% capacity). While this did reduce project headcount costs, there was a knock-on negative impact both to the schedule and other line items in the project budget.

- The 80% reduction in the Quality Control Analyst numbers created a bottleneck of 1 person for all system testing work. This meant that even when development and data engineering teams had completed their work, there was only one person available to produce test cases, automate testing scripts and run those tests.
- This also created a situation where defects were missed during testing. These defects were often found late in the development cycle - during deployment of new code, or by client staff after deployment. Instead of being found early in the cycle and fixed quickly, development and deployment cycles needed to be re-done which further consumed the capacity of the project team to move on to new features and system improvements.

1.2 AI System Gaps

AI System Architecture:
The primary issue with the overall AI system architecture relates to the historical and ongoing management of data in the system. The migration of data from the seven legacy systems in the Oracle Data Mart database to the AWS RDS PostgreSQL database has resulted in missing and incomplete data in the new system. Facility and policy updates are not being updated in the RDS database which is leading to the AI models trying to make decisions based on incorrect and incomplete data. Prisoner records are not being updated for up to 12 hours which is also leading to inappropriate decisions being made.

Without accurate training datasets, any models being trained using that data will perform poorly in real-life scenarios. Additionally, when the systems are being asked to make decisions with records that for example have incorrect facility information, then those models again will make decisions that are not correct.

Algorithm Selection
The volume of data related to the Inmate Recommender tool is very low. There are only 60,000-70,000 prisoner records and around half a million records overall related to the staff, transportation costs, facilities and transportation costs records needed to make decisions about prisoner assignment and staffing.

The selection of complex algorithms (such as Neural Networks) for these systems appears to be an attempt to improve the accuracy and client satisfaction of these systems by committing significant compute costs to achieve better outcomes. Minor improvements in accuracy have come at significant cost to AI development and cloud compute costs where alternative and much cheaper solutions such as improvements to the data, query expressiveness and quality of data labelling during model training are likely to have achieved far better results with simpler inexpensive algorithms.

Query Performance

The latency of query response in the RDS Database is sub-optimal. The database itself holds around 500,000 records with a low number of Create / Read / Update / Delete operations. Neither the volume of data nor the number of operations on that data should require significant storage or compute operations to perform well. Nonetheless, the client reports latency in processing prisoner assignment recommendations (7 seconds per recommendation), and in updating prisoner status (12 hours).

The expected results from database queries have also been a source of concern for the client as they believe there is inaccurate data and records with missing data being returned. This points to a poor data migration from the original Oracle Data Mart to the new AWS RDS database. The team responsible for the original migration experienced challenges with consistently mapping the original dataset to a new schema and replicating the query functionality of non-standard Oracle SQL features to the new PostgreSQL database management system that doesn't support those features.

Resource Allocation

In an attempt to improve the accuracy and performance of these systems, it appears that decisions have been made to consume more expensive cloud services while trying to offset these increased costs by reducing the project headcount. While on an accounting basis this might appear to be a logical decision, what it has in fact done is slowed down the project while increasing the overall cost of the project, without providing the desired performance and accuracy improvements. This is because the main root cause of the performance and accuracy issues (data quality and query performance) were not addressed and could not be overcome by using more expensive compute and storage services. The reduction in project headcount and expertise may have actually contributed to the failure to solve the data and query problems, creating a negative feedback loop that made both the system performance and cost problems worse.

1.3 Technical Analysis:

In general, applying additional memory, computation resources and parallel processing such as multithreading improve the performance of AI model training and inference, but at much higher cost. Training higher complexity models such as deep neural networks require significant computational capacity and memory than simpler models such as decision trees. Their inference environments (where the models are deployed after training to help make decisions) also require more computation and memory resources to make decisions.

Due to the complex nature of neural networks, parallel processing enabled by technologies such as GPUs that break the work down into independent units that can be processed at the same time as other work units, are usually needed to keep training and response times within acceptable boundaries. However these resources are significantly more expensive than more serialized technologies such as CPUs and should only be considered for specific problems where parallel processing costs are justified by significantly improved performance of accuracy in training and inference. In the case of the Inmate Recommender and Staffing Prediction systems, the application of GPU resources created significant cost overrun without making any significant contribution to the performance or accuracy of those systems.

It is critical when considering a trade-off of algorithmic complexity, computational resources and memory that there may be much more efficient ways of improving performance first. In the case of the prison management systems, there is a clear upstream problem with the quality of the data used for training and inference that is impossible to overcome with expensive cloud resources. Addressing the quality of the data upfront will improve the precision of the models produced and could feasibly result in the achievement of desired system performance without the need for excessive operational cloud costs.

2.1 Resource Optimization:

Optimize Memory Usage

To optimize memory usage in the training and inference environment, a number of techniques should be considered.

1. **Algorithm selection:** Selecting algorithms appropriate to the dataset and desired business outcomes will help to avoid the excessive use of expensive memory resources.
2. **Efficiency in data loading:** For training models, not every record nor every feature adds value to the training process. Loading records selectively; optimizing for the right balance between model accuracy and data training volume; and only including data features that contribute to model decision-making should be included in the training dataset. Some experimentation is required to understand and measure the relationships between decisions and specific data features. During initial interactive phases of development, smaller datasets should be used at first, with increasing dataset sizes used as the environment is tuned and improved.
3. **Categorical Data Encoding:** Converting non-numeric (categorical) variables into numeric formats can help reduce memory usage. Techniques such as one-hot encoding and label encoding can significantly reduce memory usage without compromising on model performance, but it is important to understand what kind of encoding is appropriate for different algorithms.

Optimize Query Expressiveness

Optimizing query expressiveness aims to improve the quality of queries, especially where datasets have been created from different sources that contain different representations of the same data - such as different features / columns, different data structures, different levels of data normalization and different primary and foreign keys. Datasets constructed in this way can be difficult for a human to accurately assemble into a complete and consistent dataset with the result that query results may seem incomplete or just incorrect.

Traditional SQL querying is based on relational algebra which is a procedural query language that is focused on *how* the system should respond through a sequence of joins and filters. A Categorical Query Language (CQL) on the other hand is based on relational calculus and is more focused on *what* data to retrieve by declaring the relationships between different data elements even across multiple data sources and creates an abstraction layer that hides the need to specify the series of steps needed to retrieve the desired results. This can lead to higher performance, with less risk of human error.

Considering a database management system and toolset that supports CQL can help to improve the quality and performance of queries executed in the system backend and shown to end-users.

Optimize Compute Power Usage

Align system design with actual processing needs. Neither the Inmate Recommender nor the Predictive Staffing system requires real-time responsiveness. Both systems should be shifted to batch processing: the Inmate Recommender can run daily, while Predictive Staffing runs weekly.

Additionally, the Inmate Recommender and Predictive Staffing systems should only be run on CPU-based compute environments. As the datasets are relatively small, the algorithms likely to yield the most efficient results are simpler, and the environments can run in batch, there is no benefit to using expensive parallel processing hardware such as GPUs or TPUs which would be more appropriate for more complex real-time systems.

On the other hand, a GPU or TPU is appropriate for the Security Compliance Monitoring tool as this training needs to be done with many tens of thousands of images and be able to make detections in real-time. GPUs are general parallel-processing chips that were originally designed to cater for the demands of video gaming. TPU's on the other hand are more specialized hardware components designed for complex deep learning. AWS offers the Trainium TPU which is optimized for deep learning and generative AI model training and claims higher performance and efficiency compared to more generalized GPUs. Some experimentation to understand the best selection for the Security Compliance Monitoring tool would help to assess which offers the best results within cost constraints.

2.2 AI Model Improvements:

The critical first step before any other action is taken is to assess the quality of model training, including both the data used for training the models, and the quality of the labelling process used. Poor data and poor data labelling are going to render any experimentation on model selection moot.

The Inmate Recommender and Predictive Staffing systems are both relatively small datasets that use numerical and textual data to make decisions around staffing levels and inmate housing. Much simpler and inexpensive algorithms such as Decision Tree, Random Forest or Naive Bayes algorithms are likely to yield acceptable levels of accuracy as long as the training data quality is high. Neural network and deep learning algorithms are unlikely to yield significantly better results at an acceptable cost.

Some experimentation may be needed to guide the selection of the best algorithms for each system with some trial and error with the models, configuration parameters and data selection / encoding choices.

2.3 Timeline & Budget Strategy:

The project has less than one year of the original five years left to run and is already significantly over budget and not meeting the expectations of the client. It's important that remedial actions be taken where these actions must:

- Deliver perceptible and measurable improvements in the system performance
- Be deliverable in a short timeframe
- Not further add costs that contribute to further budget overruns

For the remainder of the project term, the project team needs to firstly identify "quick wins" that meet these criteria and execute rigorously to achieve them. An initial focus on cost savings in the cloud environment can help to fund reinvestment in other areas such as project staffing. Secondly, the project needs to move into a more continuous Observe-Orient-Decide-Act (OODA) cycle of execution.

The following high-level strategy for restoring the health of the project is as follows:

Assessment Phase - 2 Weeks

This initial phase is to clearly define the main problems with the system, what may be causing those problems, and to identify the most immediate and impactful actions that can be taken quickly. The purpose of this phase is to ensure that future action plans are data-driven, and create a baseline against which the project team can measure progress over subsequent phases.

Assessment Phase Activities:

- Perform a review of AWS cloud costs by system and cloud service.
- Perform a review of the full model training lifecycle, as well as the algorithms used for training models.
- Perform a review of the data quality in the PostgreSQL database, identifying opportunities to improve the

quality of that data.
- Review quality metrics such as defect reports and customer feedback (such as tickets, satisfaction surveys) and perform a “five why” exercise to understand the root causes of these issues.
- Perform review of staffing of key roles within the project team.
- Understand why prisoner records are not being updated for up to 12 hours and identify operational procedures and controls that can address this problem.
- Initiate a weekly project governance where actions and metrics are reviewed.

Assessment Phase Outcomes:
- Understand what factors are driving the large increases in cloud costs.
- Understand what the model training pipeline is, why certain algorithms were selected over other, potentially cheaper, algorithms.
- Understand the quality of the data in the database, such as missing data, broken referential integrity and duplication of records.
- Understand how quality is measured and what factors are contributing to poor quality outcomes.
- Understand how the project is staffed relative to the work needed to complete the project satisfactorily. Is the team experiencing bottlenecks in certain areas (such as quality control) due to resource constraints? Is the project potentially over-staffed in other areas relative to the problems that need to be addressed.
- Ensure there is a governance process in place to ensure continued focus on improvements and achievement of goals, with quick decision-making to adjust course when needed.

Quick Win Phase 4 - 8 Weeks

This phase is a critical-action phase where the most impactful remedial actions identified during the Assessment Phase that can be executed quickly are done, with outcomes from these actions measured and monitored. One of the main goals of this phase is to show immediate operational cloud cost reductions that can be used to minimize the overall budget overrun as well as fund some carefully-selected investment in critical roles.

Quick Win Phase Activities:

- Make quick adjustments to cloud service usage that don’t require major development scheduling the stand-up of cloud compute capacity only when it’s needed for batch processing, downgrading compute and storage services to the minimum levels needed for the systems to operate. For example, ensure only CPU resources are used where serial processing of relatively small amounts of data is appropriate. Review GPU usage and move to cheaper and more effective TPU resources for image analysis.
- Implement tactical improvements to the performance of the PostgreSQL database for example by improving use of indexes, and optimizing bottleneck queries.
- Pause feature development to focus on reducing the number of high priority system defects.
- Implement operational procedures and controls to ensure that prisoner records are updated quickly within an acceptable time.

Quick Win Phase Outcomes:

- Immediately reduce cloud operational expenses. Negotiate with the client to use a portion of the anticipated savings to rebalance the project staff team, especially to hire 2-3 additional quality engineer contractors for the remainder of the project.
- Improve overall system performance by ensuring the PostgreSQL database is returning results more quickly without major changes to data structure.
- Improve the quality of the training data and data labelling processes to help improve the quality of the models being produced for decision-making.
- For the Security Compliance Monitoring system, because the training data is highly unbalanced towards records classified as non-suspicious, the project team should look to augment the model training process through techniques such as rebalancing the dataset used for training. Over-sampling suspicious records in the training dataset may help to improve the models ability to identify suspicious activities. Adjustments to

the dataset should be carefully monitored as there is also a cost to falsely labelling an incident as suspicious and may lead to a lack of trust in the system if there are too many false positives.
- Improve quality of prisoner records and reduce system mal-performance caused by missing or out of date prisoner records.

Ongoing Continuous Improvement Phases (10 phases X 4 Weeks Each) - OODA

For the remainder of the project, activities need to be planned on an iterative basis and based on feedback from the previous iteration. Constant monitoring of a broad range of metrics and reporting of those metrics to project stakeholders will help to ensure that the remaining time is used to focus on the most important problems and to iteratively improve the systems' performance and cost profile. Regular project steering committee meetings will help to keep stakeholders engaged and accountable for adjusting plans and budget as needed to achieve these goals.

Continuous Improvement Phase Phase Activities:

- Monitor and improve the processes for collection, preparation, and cleaning of data used for model training.
- Monitor and improve data labelling activities during model training. Use measurements such as Krippendorf's Alpha to measure the consistency of data labelling.
- Monitor and improve resource consumption metrics for each system, such as average memory usage per job.
- Monitor and improve monthly compute costs across each system and by cloud service.
- Monitor and improve average query execution time.
- Monitor and improve system performance metrics such as accuracy, precision and recall. Tie all metrics back to the business requirements and strategy and critically evaluate whether any selected metric continues to be relevant.
- Review metrics and performance and weekly steering committee meetings and adjust plans as needed.

Continuous Improvement Phase Phase Outcomes:

- Improve the consistency of data labelling by ensuring poor labellers are trained or removed from the labelling activity.
- Reduce resource consumption and costs by eliminating unnecessary services, selecting better solutions and optimizing services that are used.
- Improve query response times and the quality of results from queries.
- Improve overall system performance against the baseline of business requirements.
- Engage project sponsors and other stakeholders in providing time-critical feedback and in making decisions that measurably improve the system over the remainder of the project.

Recompete Strategy:

3.1 Client-Centered Recovery Strategy:

As we are coming towards the end of this project, the client is now beginning a retendering process for the future development and operation of the systems and exploring further opportunities where AI-driven tooling might help to improve the overall efficiency and operational effectiveness of the organization. A client-centered recovery strategy needs to focus firstly on stabilizing the current project, and secondly on restoring credibility in the project team's ability to execute on a project of this type in future engagements.

Some of the client feedback on the current project referred to a perception that the systems implemented in this project were "*built for something much bigger than we needed*" and that "*the project team seemed disconnected from our actual needs*", but that our "*solutions feel like they're tailored for a completely different type of organization... as if they're more interested in showing off fancy features than in solving our real-world issues*".

The key to rebuilding trust before a recompete strategy is to demonstrate our ability to understand the customer strategy, listen to customer problems and requirements, and to run an AI development project that is driven by that strategy and those requirements. Pivoting to this client-centric rather than technology-centric approach is critical to building trust.

Fortunately, unlike our competitors we have a year to demonstrate our ability to execute in this way. A critical way to rebuild trust in the meantime is to use that time to progress against the key issues.

Budget Overruns

- Immediately rein in costs that are driven by inefficient resource usage and over-sized cloud services.
- Implement weekly financial tracking and reporting of cloud services.

Under-Performing AI

- Focus on improving the data pipeline through at-source data quality improvement, data preparation, and labelling prior to training.
- Analyse and improve the data labelling process by tracking metrics for inter-annotator agreement.
- Implement continuous monitoring of AI performance to detect drifts in accuracy and bias.

System Complexity

- Perform experimentation, initially on smaller datasets, to assess the costs and performance of different AI algorithms for each of the three new systems. Using results from the smaller datasets, select 2-3 algorithms for further assessment using larger datasets to determine the best fit algorithms for each use case.

Latency and Query Performance

- Run query profiling and index analysis in PostgreSQL to identify quick-win opportunities for improving SQL query performance.
- Evaluate if targeted investment in a more appropriate AWS RDS instance-type would improve performance within acceptable cost parameters. Memory-optimized instances are designed for workloads that require heavy data processing.

Focusing on quick-wins early in the recovery plan will help to build client confidence in our team's ability to deliver, free up budget to invest surgically in critical areas for future improvements, and demonstrate that the project team is listening to and understands the business needs of the client.

The project team needs to immediately create a communications plan to underpin the activities over the next few weeks and months until the end of the project. This communications plan is to ensure that:

- Reportable metrics are clearly defined and collected into a project dashboard that tracks outcomes such as improving data quality, performance and accuracy. Using both leading and lagging metrics, this dashboard should track the evolution of all of these metrics as remediation actions are taken.
- There is a regular weekly project steering committee meeting where progress on metrics and actions are tracked and reported by the project manager to client stakeholders and sponsors.
- Institute a monthly feedback process consisting of both surveys and in-person feedback meetings with system users to gather qualitative and quantitative sentiment data on how the system is working. These meetings should also show how previous feedback has been addressed by the project team in order to demonstrate that we are listening to the concerns of these stakeholders and reacting appropriately.

3.2 Continuous Improvement Plan:

During the next five years, the project will move from a development posture to an operations and maintenance posture. The client's concerns will be to ensure that:

- The operational cost of the system is minimized through efficient resource usage and automation of manual processes.
- The system is maintainable so that problems and outages can quickly be identified and fixed.
- The system is adaptable in response to changes in the environment such as prison regulatory changes, AI regulatory changes, and internal organizational restructuring.
- The system is scalable as new prisons come under management or prisoner numbers increase.

The client has invested significantly in the new system to improve outcomes and reduce operational costs, so it is crucial that the system remains both functional and cost-effective to run. Continuous system enhancement needs to be driven by these client needs, and not by technology fads that don't demonstrably contribute to cost reduction or measurable material system performance improvements.

A strategy for continuous system improvement over the next few years should start with defining the operational and maintenance goals of the organization (cost optimization, system latency, system accuracy, and scalability) and implementing a continuous process for evaluating changes against these goals. The goals themselves should be reviewed on an annual or as-needed basis in response to external changes such as budgetary constraints, legal and regulatory changes and changes in the organizational strategy.

An operational steering committee should be formed with appropriate stakeholders to monitor the performance and cost of the systems, track progress against operational goals, review technical and process change proposals against those goals, and make recommendations.

Potential technical improvements to address client key concerns include:

- **Database Management System migration:** It's possible that using Amazon Aurora or a NoSQL alternative like Amazon DynamoDB may be a better and more cost-effective solution than PostgreSQL. These solutions are also fully managed, and so reduce operational costs such as system patching while reducing the threat footprint caused by unpatched systems.
- **Serverless Architecture:** Moving from dedicated compute to on-demand compute resources. Amazon EC2 instances are expensive from a compute perspective. Using on-demand serverless resources such as AWS Lambda, and AWS Fargate offer the ability to pay for only compute resources that are needed in real-time. In contrast, EC2 instances are virtual machines that consume resources and budget even when they're not being used, and rarely operate close to 100% of capacity. Migrating to a true cloud-native architecture can reduce costs significantly while also shifting the burden of maintenance and security on to the cloud provider. The downside of such a strategy is that this can create an architecture that is not easily migratable to other cloud providers and create a dependency on AWS that is difficult to break away from.

3.3 Competitive Differentiation:

Our team's approach must be to use a proven track record of system optimization and a commitment to delivering rapid improvements to differentiate our offering. By adopting the recovery and recompete strategy, we can demonstrate our commitment to a customer-centric mindset, and ensure that every decision is made with the client's needs as the driving function..

Key Strengths

- **Business-driven Development**: Our approach is to start with the business strategy and needs that our customer has, not on the technologies trending in the industry. We develop bespoke solutions tailored to our clients' specific needs to reflect the unique environments in which our clients operate.
- **Experience in System Optimization**: Our team possesses expertise in identifying bottlenecks and streamlining processes. By analyzing current systems, we pinpoint inefficiencies and implement solutions that enhance productivity and reduce costs .
- **Rapid Improvement Delivery:** We adopt an agile methodology, allowing us to quickly adapt to changing demands and implement timely solutions. This agility ensures that we can deliver improvements faster than our competitors .
- **Data-Driven Decision Making:** Our strategies are informed by robust data analysis. By continuously monitoring key metrics, we make informed decisions that drive performance and align with client objectives.

Ensuring Transparency

Transparency will continue to be integral to our approach. Being able to deliver bad news as well as good news is critical to building a foundation of trust and cooperation with our client to continuously improve and adapt where needed. We will commit to:

- **Regular Reporting:** Provide the client with detailed reports on performance metrics and project progress.
- **Open Communication:** Maintain continuous dialogue to address concerns and adjust strategies as needed.
- **Collaborative Planning:** Involve the client in the planning process to ensure alignment with their goals and expectations.

By combining our expertise in system optimization with a commitment to transparency and rapid improvement, we are confident in our ability to exceed client expectations and secure success in the re-compete.

Appendix

A-1: Budget Variance

	Years 1-4	Year 5	TOTAL
Budget	$280m	$70m	$350m
Actual (Years 1-4) & Revised (Year 5)	$350m	$60m	$410m
Variance	$70m over baseline budget (+32%)	-$10m under baseline budget (-14%)	$60m over baseline budget (+23%)
Variance Causes			
AI Development:	$20m		
Cloud Compute:	$15m		
Other Costs:	$35m		

A-2: Expected Benefits Tracking

Expected Benefit	Target Metric	Actual Metric	Impact of Missed Metrics
Improved Inmate Placement Efficiency	85% accuracy 3 second processing time	65% accuracy 12 second processing time	Inmates sent to wrong facilities; increased transport costs.
Real-Time Inmate Tracking and Records	15 minute sync	12 hour sync	Inmates being detained for longer than necessary leading to risk of lawsuit and higher staffing and accommodation costs.
Optimized Staffing and Resource Allocation	80% staffing prediction accuracy	60% staffing prediction accuracy	Operational disruptions. Increased costs through over-staffing; Safety and cost risk for under-staffing.
Enhanced Security Through Proactive Monitoring	95% detection of incidents 5 minute detection reporting	75% detection of incidents 12 minute detection reporting	Missing and delayed contraband detection.
Reduced Inmate Transport & Transfers	Reduced transport costs	Reported higher transportation costs	Increased operational costs.

A-3: Data Encoding

One-Hot Encoding

Color	Red	Green	Blue
Red	1	0	0
Green	0	1	0
Blue	0	0	1

Label Encoding

Color	Encoding
Red	0
Green	1
Blue	2