Data Analytics Principles Portfolio Report

Module Code: LD7155

Lecturer Name: Anne James

Student Name: Keaton Aggarwal

QA

April 2025

Declaration

I declare that the work presented in this report is my own work. Where other published and unpublished source materials have been used, these have been acknowledged.

Word Count: 3190

Student Name: Keaton Aggarwal

Date of Submission: 12/04/2025

Signature: Keaton Aggarwal

Before submission, remember to fill in the above declaration section fields.

EXECUTIVE SUMMARY

This portfolio report examines data analytics principles and their application to optimise cloud infrastructure resources. The report analyses server performance metrics from Linnode servers to identify cost-saving opportunities and improve resource utilisation.

The research applies machine learning clustering techniques to categorise servers based on performance metrics, identifying six distinct server clusters with distinct utilisation patterns. Statistical analysis using ANOVA F-tests revealed that cost-per-project, staging project count, and memory usage were the most significant discriminating factors between clusters.

Key findings include the identification of underutilized resources, particularly in Cluster 0, which contains 33 servers with low CPU utilization (=<0.25%) and many instances (28/33) hosting no projects while still incurring daily costs. This represents a significant consolidation opportunity. The analysis also identified potential anomalies requiring further investigation, such as a server in Cluster 3 showing extremely high CPU utilisation (165%).

The ensemble clustering approach employed multiple algorithms (KMeans, Agglomerative Clustering, Spectral Clustering, and BIRCH) to determine the optimal number of clusters, achieving a silhouette score of 0.62, indicating well-defined clusters. Visualisations created for each cluster provide insights into resource utilisation patterns, and relative costs of servers to help identify optimisation targets.

Limitations of the analysis include the use of static data captured at a single point in time. Future improvements would incorporate time-series analysis of performance metrics to provide a more comprehensive understanding of resource utilisation and efficiency patterns over a longer period.

The product is a proof of concept, demonstrating the value of applying data analytics to infrastructure management and provides a foundation for developing a streaming analytics solution that could deliver ongoing infrastructure optimisation insights, potentially reducing costs and improving resource efficiency across the organisation's cloud infrastructure.

Table of Contents

List of Figures

List of Tables

1. Business Challenge and Context

Mycelium is a web application being developed by our DevOps team that functions as a Remote Monitoring and Management (RMM) tool, collecting real-time data from our Linnode cloud infrastructure environment. It streamlines workflows by integrating multiple functions into a single interface: server metrics analysis, GitHub actions execution, Docker container deployment, and web domain management with CloudFlare.

Our organisation's cloud infrastructure currently lacks adequate analysis of key optimisation metrics. The primary issues include:

1. Resource management challenges: Service overprovisioning and underutilised compute resources contributing towards inefficient overhead spend.
2. Project lifecycle issues: Difficulty identifying infrastructure that is still required due to limited organisational knowledge of the infrastructure, which blocks the ability to decommission and rationalise the environment with informed decision making.
3. Cost transparency gaps: Limited understanding of the granular impacts of server infrastructure, efficiency and configuration on aggregate costs in the cloud infrastructure environment.

For this report, we captured a database snapshot to evaluate whether data analytics approaches can provide valuable infrastructure insights for resource optimisation. This analysis serves as a proof-of-concept for a more advanced streaming analytics solution that could deliver ongoing infrastructure optimisation insight with automated remediations.

Aims: In the short-term, DevOps at WCMC aims to improve service efficiency through appropriate provisioning of server resources that host revenue generating projects. In the long-term, the team hopes to automate processes in Continuous Delivery/Continuous Improvement using a combination of Machine Learning (ML), Artificial Intelligence (AI) and analytical tools and techniques.

Objectives: Apply data analytics techniques to transform raw infrastructure data into actionable insights to support the short-term aims of DevOps team. Develop granular metrics that measure efficiency and costs of servers and use this data to produce optimisation targets in the environment.

Methods:

1. Data extraction, transformation and analysis with Python and SQL.
2. Machine Learning using a consensus clustering ensemble to create groupings of server resources with similar patterns with respect to efficiency and cost metrics.
3. Data Visualisation with Tableau to translate quantitative insights into meaningful business intelligence for wider audiences giving a qualitative interpretation of the data.

Impact:

1. Develop the analytical approach required to advance data-driven decision-making capacity for DevOps and IT Infrastructure at UNEP-WCMC that is reproducible and scalable with respect to long-term aims.
2. Generate a set of resource optimisation targets to inform cloud infrastructure strategy in line with the short-term aims of resource and cost optimisation.
3. Provide senior management stakeholders with visual representations of cloud infrastructure to demonstrate current state of cloud infrastructure with audiences without deeper domain knowledge in digital spheres.

2. Data Analytics Principles

2.1 Key Algorithms and Models

Machine Learning (ML) is used to develop analytical solutions for data-driven decision making across myriad domains in research and business alike (Ramanathan *et al.*, 2017).
The development of ML solutions for analytics occurs in the modelling stage of the Cross Industry Standard Process for Data Mining (CRISP-DM), a framework for producing analytics products, which is helpful in documenting the phases of the analytics lifecycle. (Kelleher, 2018; Martínez-Plumed *et al*., 2021).

Analytical solutions enable the extraction of useful patterns and information from data to solve a problem, this process is known as Data Mining (DM). DM approaches are broadly categorised into Supervised and Unsupervised Learning approaches which diverge in methods depending on the analytics goal set by the data scientist (Kelleher, 2018).

Supervised Learning

Supervised Learning predicts target variables from input data, exemplified by classification algorithms for cancer detection in medical images and patient outcome prediction (Kumar, 2018; Richens *et al.,* 2020). While powerful for feature assignment and prediction, implementation of advanced ML techniques require organisations to overcome challenges such as skills gaps, computational limitations, and ethical and legal constraints (Iyelou & Paul, 2024). Moreover, understanding existing data relevant to a goal is essential for deploying analytical solutions in organisations. Care must be taken to evaluate model performance and overall suitability before implementation into decision-making processes. (Kelleher, 2018; Varoquaux & Cheplygina, 2022).

The suitability question is demonstrated by the limitations of regression algorithms. A

common goal for organisations is to reduce carbon footprints by being more energy efficient (Jayaprakash *et al*., 2021; Miśkiewicz *et al*., 2022). Linear regression would merely show a positive relationship between these two variables, without offering insight into the determination of energy efficiency. (Vetter and Schober, 2018; Sharif *et al*., 2019; Maheshwari, 2021)

Regression cannot extrapolate causality from quantitative models and renders reductive strategic inferences for carbon footprint reduction, simply put it implies: 'use less energy'. Understanding the underlying consumption features which influence carbon footprint would enhance the analysis with more variables. However, regression predictions deteriorate as the number of factors increases; outliers, co-linearity and multi-linearity contribute to greater noise and variance in the dataset, affecting the model's accuracy in estimating the target variable. (Maheshwari, 2021) In this example, clearly regression models would be an unsuitable approach to exploring data and patterns contributing to Carbon Footprint, therefore alternative approaches should be explored, such as Unsupervised Learning.

Unsupervised Learning

Unsupervised models are adept extracting the underlying patterns in data without predefined targets, revealing otherwise unknown features that contribute towards the overall pattern. (Kelleher, 2018; Maheshwari, 2021) In the context of our business problem, clustering techniques are useful to uncover groupings of servers with similar characteristics. For example, knowledge of server groups displaying similar CPU utilisation, cost and other efficiency metrics, prove valuable for decision making regarding efficiency. (Rodriguez et al, 2019; Jayaprakash et al, 2021; Maheswari 2021)

By including unsupervised learning methods in their approach, organisations can produce more detailed understanding of their data, resulting in better decision making. (Jayaprakash et al, 2021). The application of clustering algorithms, such as

K-means offer simple and effective means for data mining beyond the limitations of supervised learning. However, limitations such as sensitivity to outliers and reliance on heuristics affect clustering results and therefore subsequent decision making. (Rodriguez et al, 2019; Maheshwari 2021). In retrospect of these limitations organisations can seek to optimise their approach through the deployment of multiple data mining techniques in tandem to optimise their insights from analytics. (Kelleher, 2018; Mydhili et al, 2020, Jayaprakash, 2020)

Ensemble Learning

Ensemble approaches to data mining improve performance of algorithms by combining multiple ML approaches to account for the limitations of isoloated methods. This approach excels when dealing with complex, and large datasets, containing many non-linear relationships between variables. (Shu, 2016;Kelleher 2018; Pedersen and Olsen, 2020; Nanni et al, 2025)

For example, the hybrid ensemble integrating Deep Learning and Support Vector Machines (SVM) used by Nanni et al outperformed all existing approaches for species and genus classification. The efficacy of ensemble approaches is reinforced by Mohammed and Kora (2023) who demonstrate the consistent growth of these methods in their meta-review of ML papers published since 2014. Significantly, ensemble approaches are effective across in diverse applications and are more accessible than resource intensive deep-learning ensembles. (Mohammed and Kora, 2023, Rane et al, 2024)

2.2 Information Governance (IG)

Legal Requirements in the UK

There is a core set of UK information governance requirements legitimised through UK law that are designed to ensure that data is properly managed, protected and used ethically. The General Data Protection Regulation (GDPR) ensures data protection through legal requirements for all organisations processing personal data

of EU citizens. GDPR coexists with The UK Data Protection Act 2018 (DPA 2018), establishing the legal requirements for UK GDPR into domestic law. This legislation ensures organisations abide to 'data protection principles' as outlined in UK DPA 2018, which broadly cover the Accountability, Transparency and Accuracy of data, with fervent considerations on protection sensitive data associated with marginalised groups (Scantamburlo et al, 2019; GOV.UK, 2025). Failure to comply to requirements can result in legal penalties, such as fines. as well as damages to an organisation's reputation (Taylor et al 2024). As a result, compliance to frameworks such as ISO27001 and CyberEssentials are utilised by organisations to meet these requirements, with the added benefit of improving their resilience to cyber-attack. (ISO, 2022; NCSC,2025, GOV.UK, 2014)

Ethics of Data Management and Analytics

With the digitisation of social life, public trust in entities that store and manage must be balanced with utilisation of data for effective purpose, which is increasingly difficult with the emergence and expansion of big data ecosystems. (Fenech & Buston, 2020). Whilst there are promising applications in medicine, agriculture and epidemiology, Big Data analytics (BDA) presents a threat to the autonomy data subjects, specifically when applied at speed. (Kshetri, 2014, Kelleher 2018). In such cases, legislation and organisations processing data must uphold an ethical responsibility to use analytics benevolently (Kelleher, 2018).

Fry (2018) asserts that analytics has the power to fundamentally alter society, and with that, society must govern and utilise algorithms ethically to ensure its's benefits rather than out-source rationality to algorithms encoded with our own very human flaws and biases. Without these interventions, we risk losing the rights and agency that legislation seeks to uphold. (Fry, 2018, Scantamburlo et al, 2019; Green and Chen, 2019). Unified governance with normative principles around positive-sum-games may mitigate the social and ethical risks involved with BDA. Moreover, unified

governance could ensure that applications of DA remain consistent with shared values and goals of the global system – as highlighted by unified governance frameworks such as the Kunming-Montreal Global Biodiversity Framework (CBD, 2024). Building on this, a unified framework for global information governance and ethical application of analytics for human development, not power, seems imperative.

2.3 Data Storage Solutions and Pipelines

Given the importance of information governance, and the normative responsibilities associated with data management, organisations must choose effective solutions to store, transmit, process and analyse their data. The collection of tools should consider how the organisations goals for analytics align with the properties of said tools for different data storage and processing methods, the culmination of tools and methods is known as the organisational Data Architecture (Maheshwari, 2021).

For example, traditional approaches to data processing employ the Export, Transform, Load (ETL) method, which aggregates data from disparate sources into a centralised repository – a Data Warehouse (DW). Data is then migrated into Data Marts, where analytics takes place (Kelleher 2018; Foidl et al, 2024).
This approach is batch orientated, which increases latency of processing undermining the effectivity of analytical insights (Maheshwari 2021; Foidl et al, 2024). Moreover, ETL relies largely on intermediate storage between stages in the processing pipeline, and therefore requires the expansion storage systems placing more strain on organisational time and finances if they are to scale their data architecture.

In Contrast, Export-Load-Transform (ELT) methodologies excel in resolving issues of latency and scalability apparent in ETL methods where transformation occurs inside the central data store, a Data Lake. Organisations looking to gain a competitive advantage may benefit from this approach, particularly if applying predictive or streaming analytics methodologies given the reliance on instant transformation of

data as its produced. ELT approaches ingest and clean data in situ as structured, semi structured and unstructured data removing the requirement for intermediate storage (Nargesian et al, 2019). Whilst applications are broad, one example is the used of Data Lakes and ELT in Financial domains, where velocity of processing is key to analytical success. (Foidl et al 2024; Pendyala, 2025)

Given the ubiquity and growth of data in Data 3.0 (Lee, 2017) a departure from ETL to ELT approaches seems to be the zeitgeist for modern enterprises. Furthermore, ELT approaches are predominantly associated with cloud-native products, known for their scalability, flexibility and ease of access.

Cloud native Data Lakes (e.g., Amazon S3 Buckets and Azure Data Lakes) are cheap in comparison to maintaining and scaling on-premise storage. Moreover, cloud platforms are now beginning to offer integrated Analytics tools for Machine Learning (e.g., Apache Spark) within the Data Storage (Data Lakehouse's). The case for leverage for ELT over ETL is therefore favourable for the modern Big Data analytics enterprise looking to simplify their infrastructure and reduce their overheads (Foidl et al 2024; Plazotta & Klettke, 2024)

2.4 Data Hierarchies or Taxonomies: Methods, Benefits and Challenges

Organisations can use a data taxonomy to organise and classify their data into a structured hierarchy. A structured approach to data management facilitates data quality and subsequent improvement in analytical decision making (Liu, Zowghi and Peng, 2023). The FAIR (Findability, Accessibility, Interoperability, Reusability) principles guide the structured organisation of data, ensuring it remains discoverable and reusable across various analytical workflows (Wilkinson et al., 2021).
FAIR enhances data discoverability and accessibility, which is essential for analytics applications that rely on large datasets. The application of the FAIR taxonomy to creates comprehensive metadata enabling retrievability in computation and data pipelines (Wilkinson et al., 2024). Data Pipelines are integral to modern data systems using ML. However, they often deliver poor quality data due to data

integration, ingestion and compatibility issues compounded by poor data management (Foidl et al, 2024). The implementation of metadata management through data catalogues improves the performance of data pipelines, particularly in the scope of Data Lakes and Data Lakehouse's.

Whilst the benefits of data-driven governance and metadata management are clear, many organisations struggle to apply theoretical knowledge to their specific organisational context. (Jung et al, 2018; Biaggi & Russo, 2022). The characteristics of big data render traditional ETL processes ineffective; resources required to transform, structure and analyse data are not compatible with the velocity with which input data changes. As a result, the insights of analytics can lose relevance to new data produced in real time (Cai & Zhu 2015) and result in a loss of competitive edge in domains where this is a key requirement. The implementation of hierarchies and taxonomies improves the scalability of dataset without compromising the efficiency of queries and data retrieval associated with Big Data transactions (Kanathur et al, 2023). The knock-in effect is increased interoperability across an organisations systems allowing for improved, operational, strategic and governance decision making (Jung et al, 2018; Biaggi & Russo, 2022).

3. Product Design, Development and Evaluation

3. ELT Pipeline and Consensus Clustering

The product incorporates key stages of the CRISP-DM process into an ELT pipeline using ensembled unsupervised learning techniques. The pipeline cleans data in situ, performs feature selection, clustering and cluster evaluation.

1. ***Automated cleaning and pre-processing:***
    a. Pandas to extract numeric features, handle missing values.
    b. Features normalisation with RobustScaler.
    c. Initial feature selection and co-linearity removal with K-best and ANOVA F-statistics to rank feature importance and correlations.

2. ***Ensemble Clustering and Optimal Cluster count****:*
    a. Groups servers into clusters, sharing similar features in the dataset.
    b. The method uses four *base algorithms,* evaluating Silhouette scores (Sc) for each cluster.
    c. Each base algorithm calculates the optimal cluster count between 2-10 Clusters.
    d. Consensus clustering calculates the optimal clustering configuration from all samples across all base algorithms.

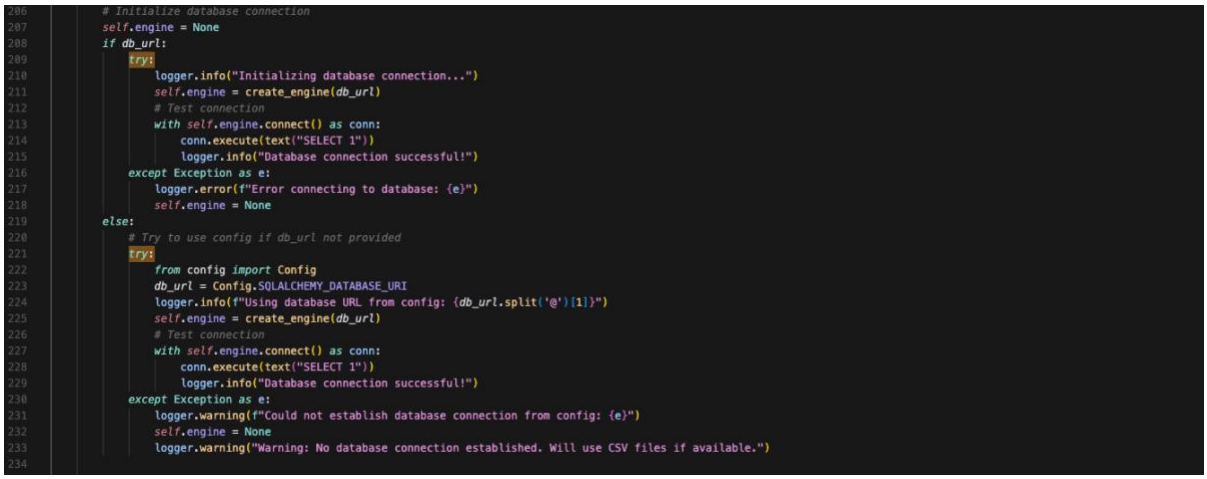
Base Algorithms and Rationales

- *K-means****:*** Used for its efficiency, scalability, and well-defined cluster centroids which aid interpretation.
- *Agglomerative Clustering:* Chosen for its hierarchical approach, which can reveal nested relationships in server resource usage without assuming spherical clusters.
- *Spectral Clustering:* Implemented to identify non-linear cluster boundaries and complex shapes in the server metric space.

- *BIRCH:* Selected for its efficiency with larger datasets and memory constraints, making it practical for ongoing server monitoring.

Data Selection and Collection

Python was used in the development of the product due to its extensive collection of ML Libraries and integration with PostgreSQL for database retrieval, transformation and processing (Hu *et al.*, 2023). It was used to create a connection to the database containing the server data. Table 1 lists the data-tables used for the analytics product and their respective features. These tables were linked to a central table through foreign keys. See Appendix C for Full entity relationship of the database.



```
        # Initialize database connection
        self.engine = None
        if db_url:
            try:
                logger.info("Initializing database connection...")
                self.engine = create_engine(db_url)
                # Test connection
                with self.engine.connect() as conn:
                    conn.execute(text("SELECT 1"))
                    logger.info("Database connection successful!")
            except Exception as e:
                logger.error(f"Error connecting to database: {e}")
                self.engine = None
        else:
            # Try to use config if db_url not provided
            try:
                from config import Config
                db_url = Config.SQLALCHEMY_DATABASE_URI
                logger.info(f"Using database URL from config: {db_url.split('@')[1]}")
                self.engine = create_engine(db_url)
                # Test connection
                with self.engine.connect() as conn:
                    conn.execute(text("SELECT 1"))
                    logger.info("Database connection successful!")
            except Exception as e:
                logger.warning(f"Could not establish database connection from config: {e}")
                self.engine = None
                logger.warning("Warning: No database connection established. Will use CSV files if available.")

```

Figure 1: PostgreSQL Database Connection using SQLalchemy

PostgresSQL Table	Infrastructure Data
wcmc_project_manager_project (Central Table in ER Strucutre - See Appendix C)	Contains project information including project name, active status, and server assignments (staging and production).
wcmc_project_manager_linodeserver	Stores the Linnode server instances with details like ID, label (server name), region, and active status.
wcmc_project_manager_servermetrics	Contains performance metrics for servers including CPU cores, memory totals, CPU load, and memory availability.

wcmc_project_manager_invoiceitem	Holds billing information with costs associated with each server.

Table 1: Data used for product development

Data Exploration and Pre-Processing

The product used SQL Alchemy to query, retrieve and join the tables in scope to create an aggregated dataset of servers and server metrics. Further metrics of efficient server usage such as daily cost per project and CPU/RAM utilisation were created. See Appendix D for the Python code to create the dataset.

44 Server records were retrieved containing 22 numeric features. Initial K-best feature selection maintained all these numeric features for clustering. 7 highly correlated features were removed from the clustering analysis.

All numeric features were extracted with any null variables filled using the median value for the feature set. These where then normalised using RobustScaler to reduce the influence of outlier data, which are common in server metrics (for example, a spike in CPU utilisation during a brief-busy period).

See Appendix E for the pre-processing code.

Clustering Results

The consensus analysis (See Appendix F) determined 6 clusters (Sc = 0.62) as the optimal cluster count for the dataset. Higher Sc indicate that clusters are well defined and within-cluster variance is low.

N_clusters (N)	Silhouette_score (Sc)	Base algorithms
6	0.62	kmeans_n6, agglomerative_n6, spectral_n6, birch_n6
2	0.60	kmeans_n2, agglomerative_n2, spectral_n2, birch_n2
3	-0.09	kmeans_n3, agglomerative_n3, spectral_n3, birch_n3
4	0.06	kmeans_n4, agglomerative_n4, spectral_n4, birch_n4

5	0.30	kmeans_n5, agglomerative_n5, spectral_n5, birch_n5
7	0.60	kmeans_n7, agglomerative_n7, spectral_n7, birch_n7
8	-0.09	kmeans_n8, agglomerative_n8, spectral_n8, birch_n8
9	0.07	kmeans_n9, agglomerative_n9, spectral_n9, birch_n9
10	0.53	kmeans_n10, agglomerative_n10, spectral_n10, birch_n10

Table 2: Consensus Clustering Silhouette Scores[1]

The analysis highlighted 6 statistically significant features in cluster separation using the ANOVA F-test. ANOVA tests whether features differ significantly between clusters by comparing within-cluster variance and between-cluster variance, with F being the ratio between the two.

cost-per-project (F = 121.44) is the most influential feature in determining cluster separation. Cost per project shows wide variance across clusters (Table 4) which is statistically significant (*P-value = 0.0000 4 d.p.)* Statistically significant P-values suggest the probability of observing differences between clusters for the given feature.

Feature Selected	F-Score (2 d.p)	P- value (4 d.p)	Significance Level
Cost per Project	121.44	0.0000	Highly Significant
Count of Staging Projects	88.38	0.0000	Highly Significant
Average Project Memory Usage	67.91	0.0000	Highly Significant
Average Project CPU Load	20.46	0.0000	Highly Significant
Production Projects	3.77	0.0072	Highly Significant
CPU Utilisation	3.37	0.0128	Significant
Cost per GB of Memory	2.10	0.0872	Not Significant
Memory Utilisation	1.62	0.1795	Not Significant

Table 3: Feature Importance in Cluster Separation - ANOVA-F [2]

[1] Table **Error! Main Document Only.**: Consensus Clustering Silhouette Scores (2 d.p.) (-1 =< Sc =< 1)

[2] Table **Error! Main Document Only.**: Feature Importance - ANOVA-F[2] (2 d.p) (p =< 0.01 = highly significant, 99% confidence level)

Product Evaluation and Suitability

Cluster	Cost Per Project (Sum)	Cost per project (Mean)	Project Count (SUM)	CPU Utilisation (Mean)	Staging Project (Count)	Production Project (Count)	Servers (Count)
0	24.24	4.85	21	0.25	17	4	33
1	11.69	2.92	11	0.33	3	8	4
2	51.71	17.24	4	0.13	3	1	3
3	0.34	0.34	36	1.65	36	0	1
4	2.62	1.31	25	0.55	25	0	2
5	31.14	31.14	1	0	1	0	1
Totals	121.82	9.63	98	0.58	85	13	44

Table 4: Clustering Data Summary

The clustering approach gives clear resource optimisation targets in line with the Business Challenge by illustrating patterns of server utilisation and targets for optimisation. Moreover, the analysis produced comprehensive data for further analysis and optimisation targeting explored further in Section 3.2.

The largest cluster (Cluster=0, Servers=33) has generally low CPU Utilisation (Mean=0.25%) with 0.64 projects per server. Many servers in this cluster host zero projects (28/33), whilst still accumulating a daily cost, representing a consolidation opportunity.

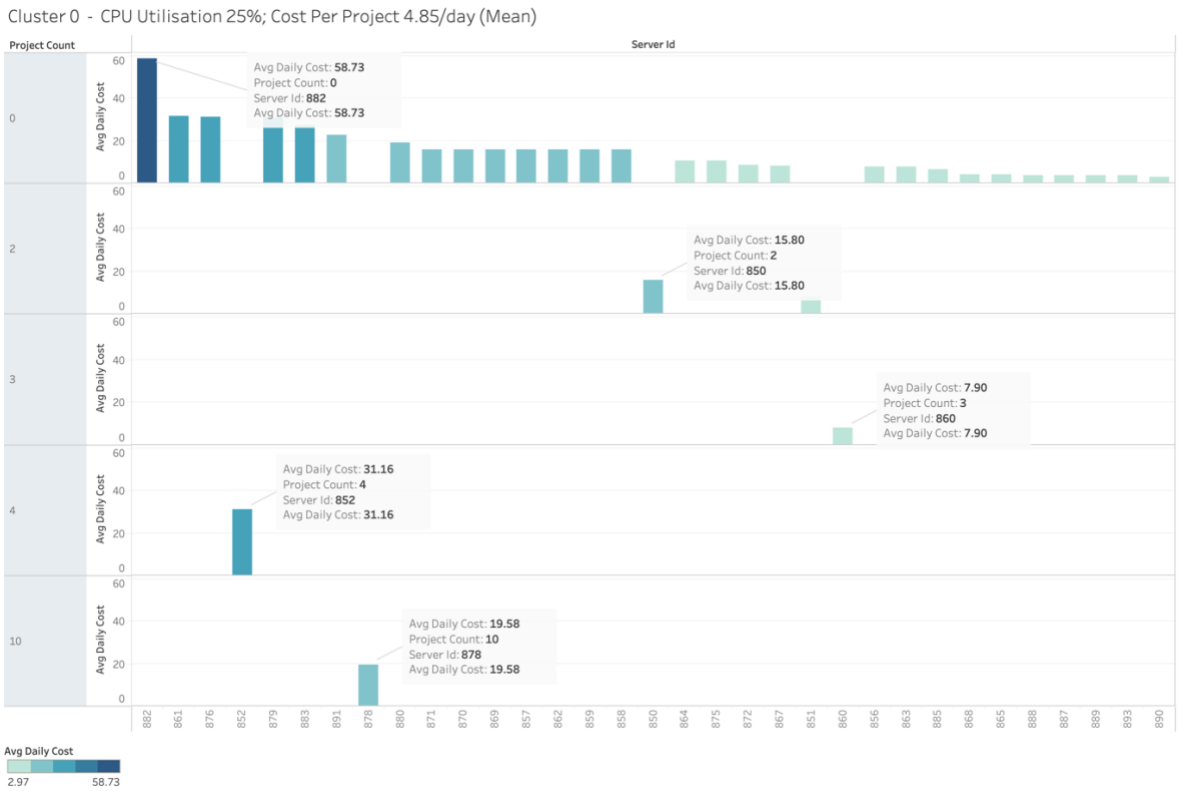


Figure 2: Average Daily Cost as an attribute for each Server Id broken down by Project Count. Colour shows details about Average Daily Cost. The data is filtered on Cluster 0.

Additional outlier detection may have improved clustering results, however Outlier inclusion in the analysis was beneficial for from a server monitoring and remediation perspective. 165% CPU Utilisation in Cluster 3 (Server Count = 1)(Project Count = 36) is a potential outlier across the dataset, however, its inclusion illustrates that the server is over-burdened and would benefit from additional computation resources. Under provisioning and failure would have an impact on the delivery of projects. With Cluster 3 in mind, server utilisation metrics were taken from a single point in time, so the outlier may be misrepresentative of real performance. For more robust results a time-series analysis with a larger sample of performance metrics would be advised.

3.2 Data Visualisation

Feature Importance in Cluster Separation

Using a bar chart to visualise feature importance score allows the summary and comparison of features and their statistical significance in a manner that is more interoperable than the raw tabular data (Kelleher, 2018).

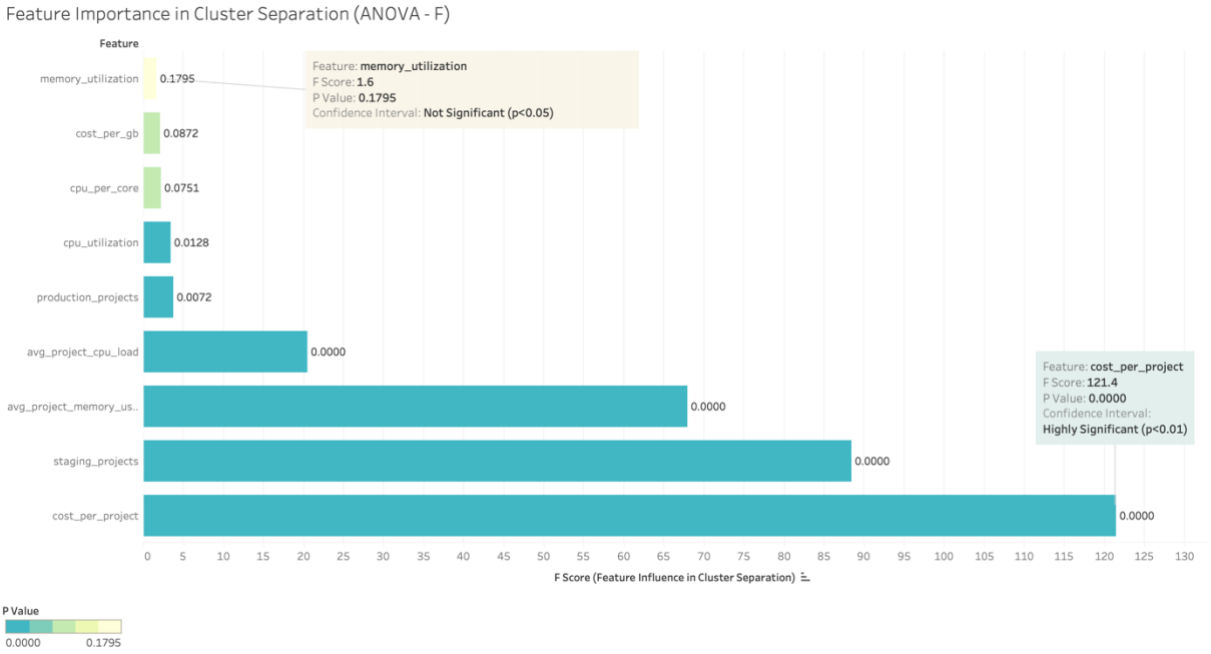


Figure 3: Bar chart of ANOVA-F Score for each Feature. Colour shows sum of P Value. Bars are labelled with P Value for Feature.[3]

A colour legend coded to the p-value aids the viewer to identify relative feature significance, with features in ascending order to create a sense of linearity. The title combined with label annotations gives a qualitative explanation of the data visualised. Labels highlighting statistical scores at the extremes of the data reinforce the understanding that larger p-values are more significant in determining cluster size. (Kelleher 2018; Maheshwari, 2021)

[3] Figure **Error! Main Document Only.**: Bar chart of ANOVA-F Score for each Feature. Colour shows sum of P Value. Bars are labelled with P Value for Feature. Confidence Intervals: Statistically significant features ($p < 0.05$), Highly significant features ($p < 0.01$)

Silhouette Scores of Consensus Clustering

A line chart was used with a drop line was used to display the optimal number of clusters in the Consensus Clustering. Gradient colour coding and annotations are with the same approach as above to make the visualisation intuitive to interpretate. The dropline emphasises the optimal result by connecting the data point to the independent variable on the x-axis. The title describes a general inference from the statistical data assisting those unfamiliar with Machine Learning terminology and the applied statistical method. (Kelleher 2018; Maheshwari, 2021)

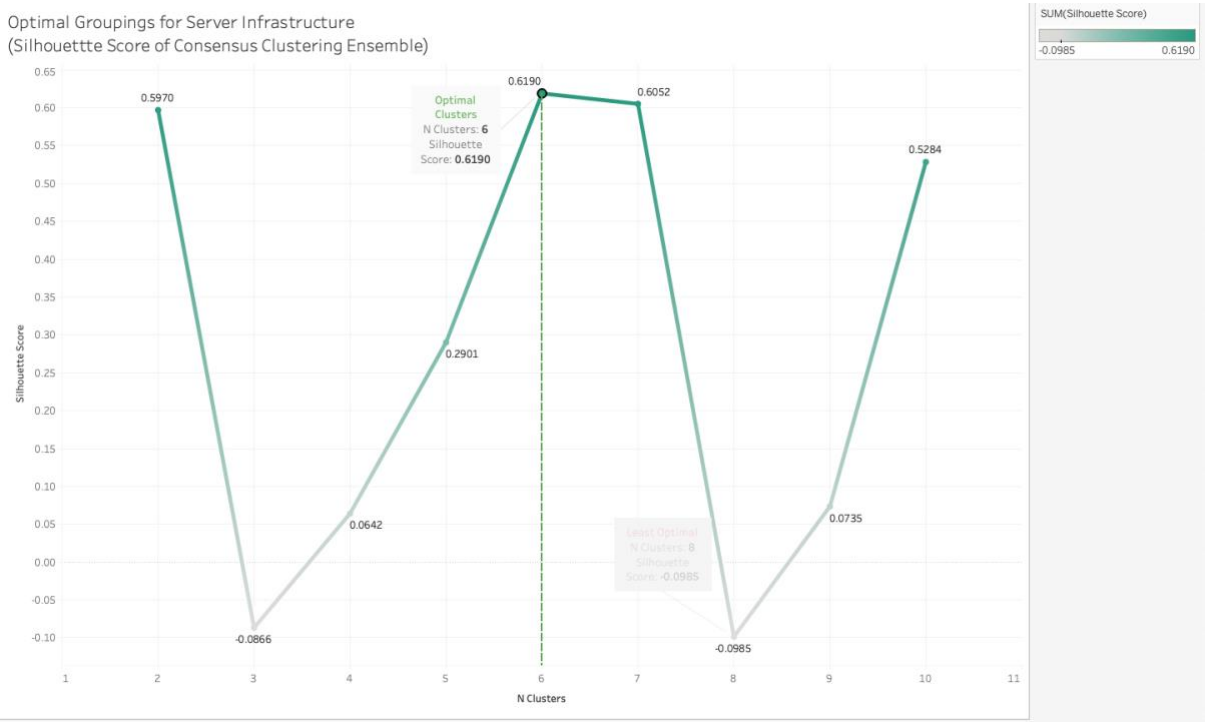


Figure 4: Optimal Groupings for Server Infrastructure displayed with a Line Chart and drop-line to emphasise the optimal clustering configuration from the Silhouette Score.

Cluster Dashboards

Dashboards are used to highlight and compare the intra-clusters trends with insights into the archetype of servers and their respective features. Pie charts group servers by their project count, with each pie segment representing a server. The additional size dimension representing that pie's monthly overhead allows comparison between the groups to reinforce the understanding of spending.

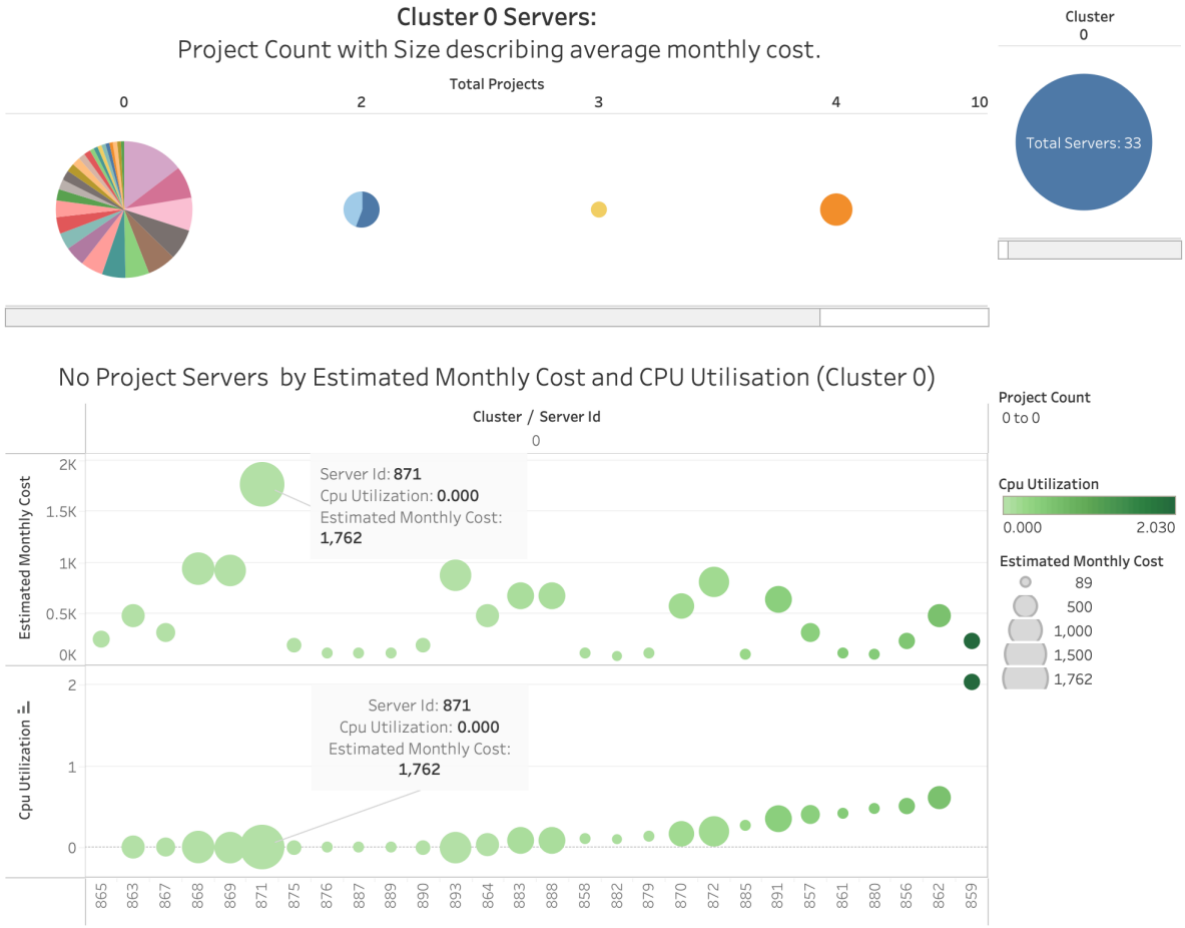


Figure 5: *Cluster 0*; Comparative Dashboard showing variance of project cost and non-linear relationships between server features.

The Dashboard drills deeper into the features of high cost, no project servers by visualising the estimated monthly cost in direct comparison with the CPU utilisation. Servers are plotted on the X-axis with instances stacked directly above each other allowing the reader to easily compare the different features. This shows various

anomalies such as intra group of servers with low utilisation and high cost, and one server with high relatively low cost, but extreme utilisation.

Figure 6 shown directly after provides a wider narrative of the Utilisation, Monthly cost and Project Counts across clusters. The axis and scales were normalised to the same scale, so the visual representation of plot size reflects the accuracy of the data.

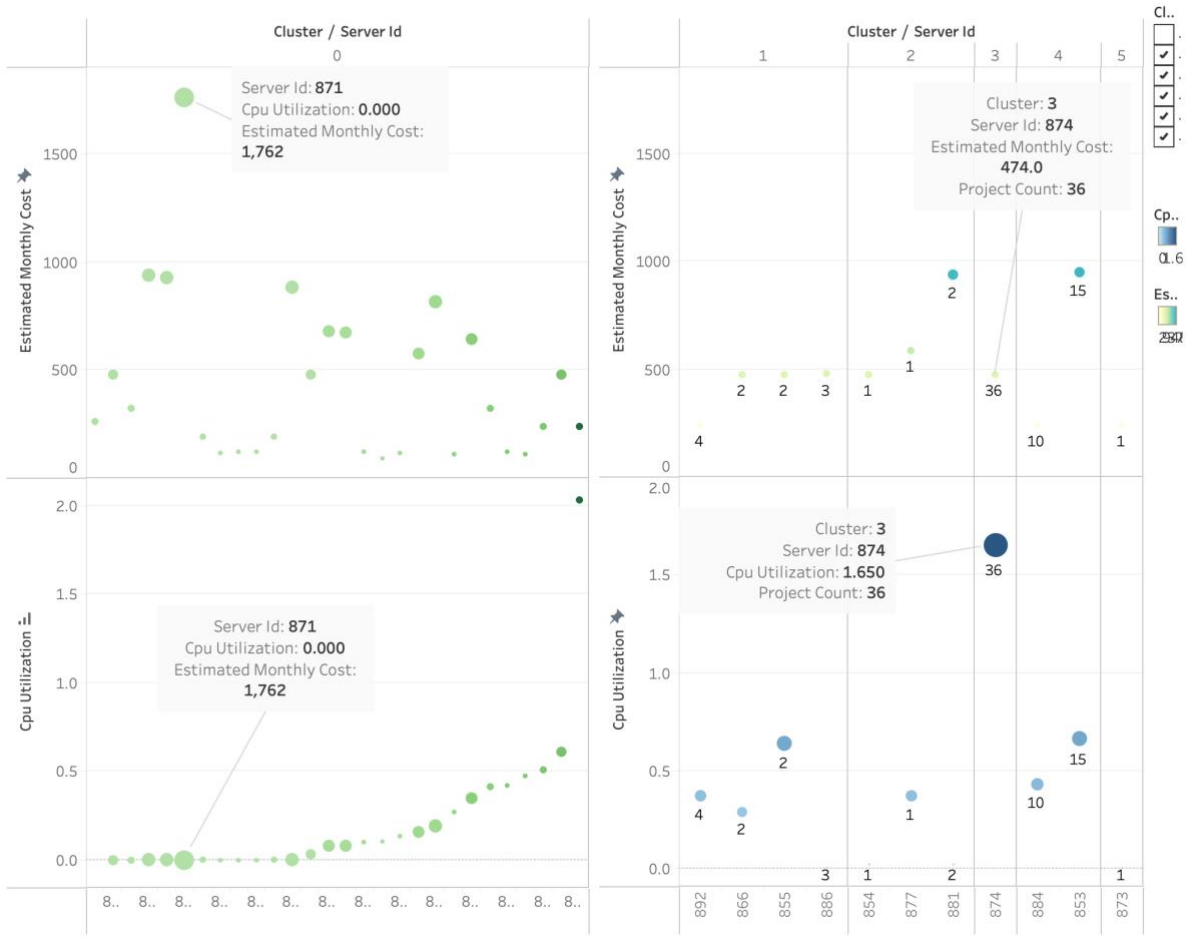


Figure 6: Clusters 1-5: Costs, Project Counts and Utilisation per server, per respective cluster.

4. Personal Reflection

The exploration of data analytics theory in Chapter 1 provided essential insights into the benefits and challenges of analytics implementations. This theoretical foundation enabled critical evaluation of appropriate analytics approaches for the business problem, informing pragmatic decisions about data architecture, hierarchy, and governance. This evaluative approach was crucial for creating a tailored solution that addresses the unique aspects of the problem beyond general resource optimisation theories and studies available in the literature, developing my skills in Data Analytics in line with the required skills of leaders in the apprenticeship standard.

Whilst not without its limitations the clustering analysis product provides valuable insights and lays the proof-concept groundwork for a more advanced analytics product for resource optimisation of Linnode Infrastructure. The product successfully categorised servers from raw data and determined key features in group classification as well as identifying cost inefficiencies through underutilised server instances.

However, the current analysis uses small sample of static data, therefore a time-series analysis with more data on server metrics should be used in the future to give more robust results. In addition, automated recommendations for optimal CPU and Memory allocation based on a time series analysis would assist improve the utility of the product.

References

1. Abdul-Azeez, O., Ihechere, A.O. and Idemudia, C. (2024) 'Enhancing business performance: The role of data-driven analytics in strategic decision-making', *International Journal of Management & Entrepreneurship Research*, 6(7), pp. 2066-2081. doi: 10.51594/ijmer.v6i7.1257.
2. Cai, L. and Zhu, Y. (2015) 'The challenges of data quality and data quality assessment in the big data era', *Data Science Journal*, 14, pp. 2-2.
3. Chan, J.Y.L. et al. (2022) 'Mitigating the multicollinearity problem and its machine learning approach: a review', *Mathematics*, 10(8), p. 1283.
4. Convention of Biological Diversity (CBD) (2024), *2050 Vision and 2030 Mission*. Available at: https://www.cbd.int/gbf/vision (Accessed: 13th April 2025).
5. Dormann, C.F. et al. (2013) 'Collinearity: a review of methods to deal with it and a simulation study evaluating their performance', *Ecography*, 36(1), pp. 27-46.
6. Fenech, M.E. and Buston, O. (2020) 'AI in cardiac imaging: A UK-based perspective on addressing the ethical, social, and political challenges', *Frontiers in Cardiovascular Medicine*, 7, p. 54.
7. Foidl, H. et al. (2024) 'Data pipeline quality: Influencing factors, root causes of data-related issues, and processing problem areas for developers', *The Journal of Systems and Software*, 207, p. 111855. doi: 10.1016/j.jss.2023.111855.
8. Fry, H. (2018) *Hello world: how to be human in the age of the machine*. London: Random House.
9. Green, B. and Chen, Y. (2019) 'The principles and limits of algorithm-in-the-loop decision making', *Proceedings of the ACM on Human-Computer Interaction*, 3, pp. 1-24. doi: 10.1145/3359152.
10. Hu, H. et al. (2023) 'Database-integrated machine learning for enhanced performance', *2023 IEEE 3rd International Conference on Big Data and Intelligent Analytics (ICBDIA)*, pp. 203-209. doi: 10.1109/bigdia60676.2023.10429411.
11. ISO (2022) *ISO/IEC 27002:2022*. Available at: https://www.iso.org/standard/27001 (Accessed: 10 April 2025).
12. Iyelolu, T.V. and Paul, P.O. (2024) 'Implementing machine learning models in business analytics: challenges, solutions, and impact on decision-making', *World*

Journal of Advanced Research and Reviews, 22(3), pp. 1906-1916. doi: 10.30574/wjarr.2024.22.3.1959.

13. Jayaprakash, S. et al. (2021) 'A systematic review of energy management strategies for resource allocation in the cloud: clustering, optimization and machine learning', *Energies*, 14(17), p. 5322. doi: 10.3390/en14175322.

14. Jung, H. et al. (2018) 'Data-driven decision-making processes, data services and applications for global aviation safety', *International Journal of Aviation Technology, Engineering and Management*, 2018(2), pp. 49-57.

15. Kabir, M.A. et al. (2024) 'Python for data analytics: a systematic literature review of tools, techniques, and applications', *Asian Journal of Science, Technology & Engineering Management Excellence*, 4(4). doi: 10.69593/ajsteme.v4i04.146.

16. Kanathur, R. et al. (2023) 'In-memory depth-first tree construction of hierarchical data in a RDBMS', *2023 7th International Conference on Computation System and Information Technology for Sustainable Solutions (CSITSS)*, pp. 1-5.

17. Kshetri, N. (2014) 'The emerging role of big data in key development issues: Opportunities, challenges, and concerns', *Big Data & Society*, 1(2), p. 2053951714564227.

18. Kumar, V.H. (2018) 'Python libraries, development frameworks and algorithms for machine learning applications', *International Journal of Engineering Research and Technology*, 7(4). Available at: https://www.ijert.org/research/python-libraries-development-frameworks-and-algorithms-for-machine-learning-applications-IJERTV7IS040173.pdf.

19. LastPass (2024) *Explained: the LastPass hack December 2024 update*. Available at: https://www.halborn.com/blog/post/explained-the-lastpass-hack-december-2024-update (Accessed: 10 April 2025).

20. Lee, I. (2017) 'Big data: Dimensions, evolution, impacts, and challenges', *Business Horizons*, 60(3), pp. 293-303.

21. Lever, J., Krzywinski, M. and Altman, N. (2016) 'Points of significance: logistic regression', *Nature Methods*, 13(7), pp. 541-542. doi: 10.1038/NMETH.3904.

22. Liu, C., Zowghi, D. and Peng, G. (2023) 'A taxonomy of factors influencing data quality', in *Data quality in big data era*, pp. 328-347. doi: 10.1007/978-3-031-34668-2_22.

23. Lomas, E. (2020) 'Information governance and cybersecurity: framework for securing and managing information effectively and ethically', in *Cybersecurity, privacy and freedom protection in the connected world*. Boca Raton: Auerbach Publications, pp. 109-130. doi: 10.1201/9781003042235-6.

24. Martínez-Plumed, F. et al. (2021) 'CRISP-DM twenty years later: from data mining processes to data science trajectories', *IEEE Transactions on Knowledge and Data Engineering*, 33(8), pp. 3048-3061. doi: 10.1109/TKDE.2019.2962680.

25. Miśkiewicz, R. et al. (2021) 'Energy efficiency in the industry 4.0 era: attributes of teal organisations', *Energies*, 14(20), p. 6776.

26. Mohammed, A. and Kora, R. (2023) 'A comprehensive review on ensemble deep learning: opportunities and challenges', *Journal of King Saud University-Computer and Information Sciences*, 35(2), pp. 757-774.

27. Mydhili, S.K. et al. (2020) 'Machine learning based multi scale parallel K-means++ clustering for cloud assisted internet of things', *Peer-to-Peer Networking and Applications*, 13, pp. 2023-2035.

28. Nanni, L. et al. (2025) 'Advancing taxonomy with machine learning: a hybrid ensemble for species and genus classification', *Algorithms*, 18(2), p. 105. doi: 10.3390/a18020105.

29. Nargesian, F. et al. (2019) 'Data lake management: challenges and opportunities', *Proceedings of the VLDB Endowment*, 12(12), pp. 1986-1989.

30. National Cyber Security Centre (no date) *Cyber essentials*. Available at: https://www.ncsc.gov.uk/cyberessentials/overview (Accessed: 10 April 2025).

31. Pedersen, C.B. and Olsen, L.R. (2020) 'Algorithmic clustering of single-cell cytometry data-how unsupervised are these analyses really?', *Cytometry Part A*, 97(3), pp. 219-221. doi: 10.1002/CYTO.A.23917.

32. Pendyala, S.K. (2025) 'Data engineering at scale: streaming analytics with cloud and Apache Spark', *Journal of Artificial Intelligence and Machine Learning*, 3(1), pp. 1-9.

33. Plazotta, M. and Klettke, M. (2024) 'Data architectures in cloud environments', *Datenbank-Spektrum*, 24(3), pp. 243-247.

34. PyPI (2025) *SQLAlchemy*. Available at: https://pypi.org/project/SQLAlchemy/ (Accessed: 10 April 2025).

35. Ramachandranpillai, R., Baeza-Yates, R. and Heintz, F. (2025) 'FairXAI-A taxonomy and framework for fairness and explainability synergy in machine learning', *IEEE Transactions on Neural Networks and Learning Systems*.

36. Ramanathan, R. et al. (2017) 'Adoption of business analytics and impact on performance: a qualitative study in retail', *Production Planning & Control*, 28, pp. 985-998. doi: 10.1080/09537287.2017.1336800.

37. Rane, N., Choudhary, S.P. and Rane, J. (2024) 'Ensemble deep learning and machine learning: applications, opportunities, challenges, and future directions', *Studies in Medical and Health Sciences*, 1(2), pp. 18-41.

38. Ranglani, H. (2024) 'Comparative analysis of clustering algorithms on synthetic circular patters data', *Machine Learning and Applications*, 11(4), pp. 17-30. doi: 10.5121/mlaij.2024.11402.

39. Richens, J.G., Lee, C.M. and Johri, S. (2020) 'Improving the accuracy of medical diagnosis with causal machine learning', *Nature Communications*, 11(1), p. 3923.

40. Rodriguez, M.Z. et al. (2019) 'Clustering algorithms: a comparative approach', *PloS One*, 14(1), p. e0210236.

41. Scantamburlo, T., Charlesworth, A. and Cristianini, N. (2019) 'Machine decisions and human consequences', *arXiv: Computers and Society* [Preprint]. Available at: https://dblp.uni-trier.de/db/journals/corr/corr1811.html#abs-1811-06747.

42. Sharif, A. et al. (2019) 'The dynamic relationship of renewable and nonrenewable energy consumption with carbon emission: a global study with the application of heterogeneous panel estimations', *Renewable Energy*, 133, pp. 685-691.

43. Shu, H. (2016) 'Big data analytics: six techniques', *Geo-spatial Information Science*, 19(2), pp. 119-128. doi: 10.1080/10095020.2016.1182307.

44. Taylor, J., Henriksen-Bulmer, J. and Yucel, C. (2024) 'Privacy essentials', *Electronics*, 13(12), p. 2263. doi: 10.3390/electronics13122263.

45. UK Government (2024) *Cyber essentials scheme: overview*. Available at: https://www.gov.uk/government/publications/cyber-essentials-scheme-overview (Accessed: 10 April 2025).

46. UK Government (no date) *Data protection*. Available at: https://www.gov.uk/data-protection (Accessed: 10 April 2025).

47. Varoquaux, G. and Cheplygina, V. (2022) 'Machine learning for medical imaging: methodological failures and recommendations for the future', *npj Digital Medicine*, 5, p. 48. doi: 10.1038/s41746-022-00592-y.

48. Vetter, T.R. and Schober, P. (2018) 'Regression: the apple does not fall far from the tree', *Anesthesia & Analgesia*, 127(1), pp. 277-283. doi: 10.1213/ANE.0000000000003424.

49. Wilkinson, S. et al. (2024) 'Applying the FAIR principles to computational workflows', *arXiv* [Preprint]. doi: 10.48550/arxiv.2410.03490.

Appendix A: Pathway Standards Addressed

Standard	Evidence
TK1) How key algorithms and models are applied in developing analytical solutions and how analytical solutions can deliver benefits to organisations	In Chapter 2.1 I discussed how machine learning algorithms are applied through the CRISP-DM Process for data driven decision making, particularly through data mining Organisational data for extraction of insightful information. I outlined the utility and application of the main paradigms of Machine Learning (Supervised Learning and Unsupervised Learning)) as well as discussing the strengths and limitations of related ML methods in each scope both theoretically and in application of business problems. I highlighted how Ensemble ML Based approaches are incumbent across many academic domains due to their efficacy when compared to singular ML approaches for Data Analytics.
TK2) The information governance requirements that exist in the UK, and the relevant organisational and legislative data protection and data security standards that exist. The legal, social and ethical concerns involved in data management and analysis	The core set of legal requirements such as UK DPA and UK GDPR were discussed in Section 2.1- Legal requirements in the UK. I highlighted the principles of data protection in the relevant laws and the legal implications and penalties of non-compliance for organisations. I discussed how organisations can leverage international and domestic information governance frameworks such as ISO27001 and CyberEssentials to enhance Cyber Security and ensure compliance with the legal requirements for Data Protection and Security. In 2.1-Ethics of data management and Analytics I highlighted the ethical concerns of Big data analytics on individual autonomy particularly when applied at speed. I highlighted how comprehensive Governance is required mitigate the risks that Data Analytics is used ethically. Specifically, applications should seek to

	minimise risks to the rationality and agency of human decision making. In a wider context I discussed how Data Analytics has the capacity to transform society at every level. I asserted that the use of Data Analytics should have a benevolent normative foundation for Human Development as its principle for application rather than the advancement of individual interests in Zero-sum-games.
TK3) The principles of data driven analysis and how to apply these. Including the approach, the selected data, the fitted models and evaluations used to solve data problems	Chapter 2.1 highlighted the principles of data driven analytics, discussing the suitability of algorithms are for different approaches in business insight generation from data. The understanding of theory was used to drive the approach taken in 3.1 to work towards the objectives of the business challenge. The approach incorporated the understanding that Ensemble Approaches to ML provide more robust results for Machine Learning. The ensemble approach itself include an evaluative mechanism to optimise modelling results via the use of Silhouette Score. The selected dataset for model development was chosen using Data Analytics techniques for data cleaning, combines with the understanding that this data set would generate meaningful insight with respect to solving the business challenge. The product - an automated ELT process streamlines the CRISP-DM by incorporating key stages into a Python script to programmatically draw insight from company data.
TK4) The properties of different data storage solutions, and the transmission, processing and analytics of data from an enterprise system perspective. Including the platform choices available for designing and	Different Data Processing Techniques are discussed in Chapter 2.3, as well as the properties, pros/cons of different storage solutions. The suitability of systems, tools and processes for. Data Analytics is considered against the approach required by

implementing solutions for data storage, processing and analytics in different data scenarios	the organisation. Theoretical recommendations were made based on the stream analytics approaches to data analytics. This approach aligns with the long-term product development goals of my organisation, which is highlighted in the Business Challenge context.
TK5) How relevant data hierarchies or taxonomies are identified and properly documented	Chapter 2.4 Discusses this aspect of the apprenticeship standard. Theoretical benefits, challenges and methods related to the knowledge area were highlighted and related to the Data Analytics approach of ELT Data Pipelines, demonstrating the depth theoretical knowledge in relation to the Business Challenge and Product Development.
TK6) The concepts, tools and techniques for data visualisation, including how this provides a qualitative understanding of the information on which decisions can be based	Chapter 3.2 Explores the results of the product through the applied use of Data Visualisation Theory. Key quantitative results were visualised in Figures to give a qualitative understanding of the data for onward decision making. The theory of visualisation was applied to each Figure, with a discussion and interpretation for the visualisation. Academic rationales for the tools and techniques were included in the discussion for each figure to demonstrate its suitability. See Feature Importance in Cluster Separation for an example of visualisation figures and related discussions on this knowledge area.

Appendix B: Feature Selection

```
# 1. Extract numeric features and handle missing values
    numeric_df = df.select_dtypes(include=['number'])
    numeric_df = numeric_df.fillna(numeric_df.median())

    # Store feature names
    feature_names = numeric_df.columns.tolist()
    logger.info(f"Working with {len(feature_names)} numeric features")

    # 2. Scale the data
    if scaler_type == 'standard':
        scaler = StandardScaler()
    elif scaler_type == 'robust':
        scaler = RobustScaler()
    else:
        logger.warning(f"Unknown scaler type: {scaler_type}. Using RobustScaler.")
        scaler = RobustScaler()

    scaled_data = scaler.fit_transform(numeric_df)

    # 3. Feature selection
    selected_data = scaled_data
    selected_feature_names = feature_names

    if feature_selection:
        if feature_selection == 'variance':
            # Remove low-variance features
            selector = VarianceThreshold(threshold=0.01)
            selected_data = selector.fit_transform(scaled_data)
            mask = selector.get_support()
            selected_feature_names = [feature_names[i] for i in range(len(feature_names)) if mask[i]]

            logger.info(f"Selected {len(selected_feature_names)} features using variance threshold")
```

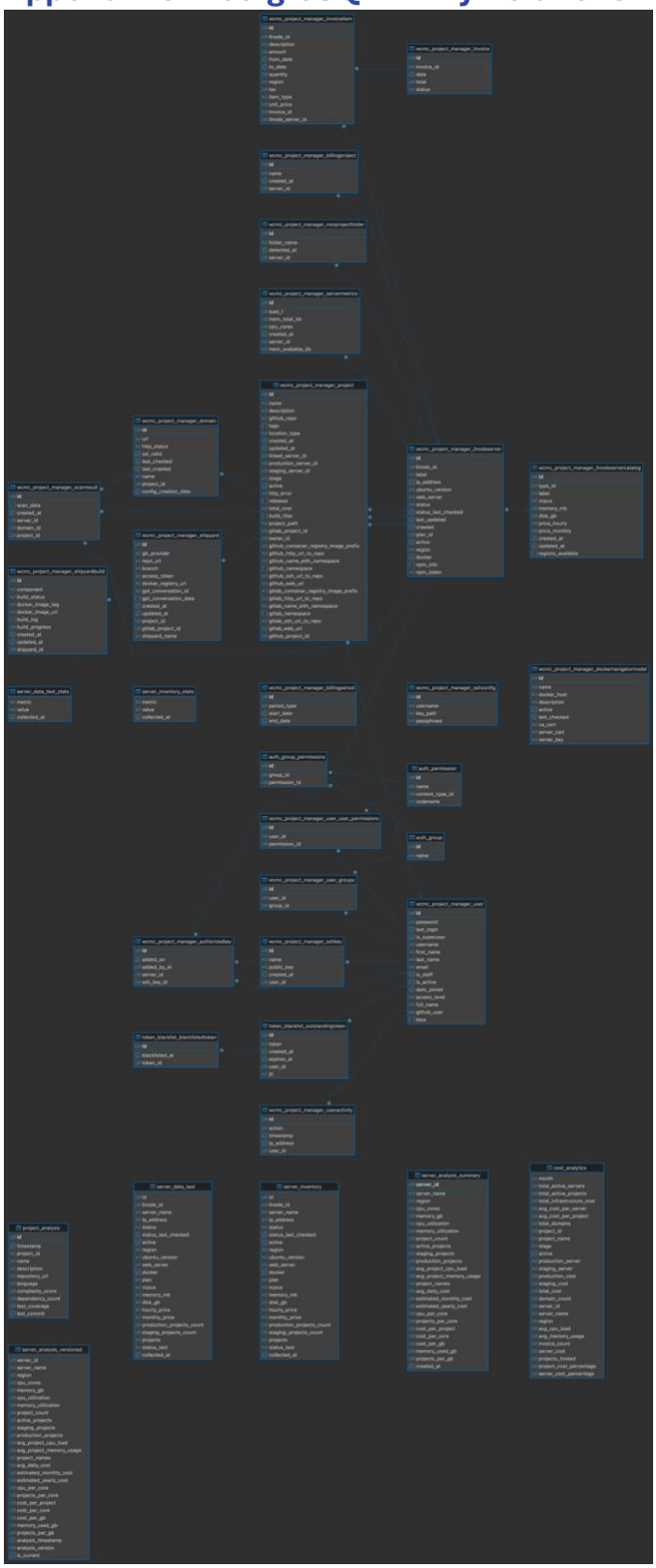
```
    elif feature_selection == 'kbest':
        # First cluster the data to get pseudo-labels
        kmeans = KMeans(n_clusters=min(5, len(df) - 1), random_state=42, n_init=10)
        pseudo_labels = kmeans.fit_predict(scaled_data)

        # Select k best features based on ANOVA F-statistic
        k = max(int(len(feature_names) * 0.7), 5)  # Select 70% of features or at least 5
        selector = SelectKBest(f_classif, k=k)
        selected_data = selector.fit_transform(scaled_data, pseudo_labels)
        mask = selector.get_support()
        selected_feature_names = [feature_names[i] for i in range(len(feature_names)) if mask[i]]

        # Get feature scores
        scores = selector.scores_
        feature_scores = list(zip(feature_names, scores))
        sorted_features = sorted(feature_scores, key=lambda x: x[1], reverse=True)

        # Plot feature importance
        plt.figure(figsize=(12, 8))
        selected_scores = [score for name, score in sorted_features if name in selected_feature_names]
        selected_names_sorted = [name for name, _ in sorted_features if name in selected_feature_names]

        plt.barh(range(len(selected_scores)), selected_scores, align='center')
        plt.yticks(range(len(selected_scores)), selected_names_sorted)
        plt.xlabel('F-Score')
        plt.title(f'Top {k} Features by F-Score')
        plt.tight_layout()
        plt.savefig(os.path.join(self.output_dir, 'feature_importance.png'))
        plt.close()

        logger.info(f"Selected top {k} features using k-best method")

# 4. Check for highly correlated features
corr_matrix = numeric_df[selected_feature_names].corr()
high_corr_pairs = []
```

```
    for i in range(len(selected_feature_names)):
        for j in range(i+1, len(selected_feature_names)):
            if abs(corr_matrix.iloc[i, j]) > 0.8:
                high_corr_pairs.append((
                    selected_feature_names[i],
                    selected_feature_names[j],
                    corr_matrix.iloc[i, j]
                ))

    if high_corr_pairs:
        logger.info(f"Found {len(high_corr_pairs)} highly correlated feature pairs")

        # Sort correlation pairs by absolute correlation value
        high_corr_pairs.sort(key=lambda x: abs(x[2]), reverse=True)

        # Create a sorted correlation matrix for visualization
        # First, create a list of all features involved in high correlations
        all_corr_features = set()
        for f1, f2, _ in high_corr_pairs:
            all_corr_features.add(f1)
            all_corr_features.add(f2)

        # Create a subset of the correlation matrix with only highly correlated features
        corr_subset = corr_matrix.loc[list(all_corr_features), list(all_corr_features)]

        # Create a figure with two subplots
        plt.figure(figsize=(15, 10))

        # Plot 1: Sorted correlation matrix
        plt.subplot(1, 2, 1)
        mask = np.triu(np.ones_like(corr_subset, dtype=bool))
        sns.heatmap(corr_subset, mask=mask, cmap='coolwarm', center=0,
                    annot=True, fmt='.2f', square=True, linewidths=.5)
        plt.title('Highly Correlated Features Matrix')

        # Plot 2: Bar plot of correlation values
        plt.subplot(1, 2, 2)
```

```
        # Extract correlation values and feature pairs
        corr_values = [abs(pair[2]) for pair in high_corr_pairs]
        feature_pairs = [f"{pair[0]} vs {pair[1]}" for pair in high_corr_pairs]

        # Create bar plot
        bars = plt.barh(range(len(corr_values)), corr_values)
        plt.yticks(range(len(corr_values)), feature_pairs)

        # Add correlation values as labels
        for i, bar in enumerate(bars):
            plt.text(bar.get_width(), bar.get_y() + bar.get_height()/2,
                     f'{high_corr_pairs[i][2]:.3f}',
                     va='center', ha='left')

        plt.title('Feature Correlation Values')
        plt.xlabel('Absolute Correlation')

        plt.tight_layout()
        plt.savefig(os.path.join(self.output_dir, 'feature_correlation_analysis.png'))
        plt.close()

        # Log the top 5 most correlated pairs
        logger.info("\nTop 5 most correlated feature pairs:")
        for f1, f2, corr in high_corr_pairs[:5]:
            logger.info(f"  {f1} and {f2}: {corr:.3f}")

        # Create a CSV file with all correlation pairs
        corr_df = pd.DataFrame(high_corr_pairs, columns=['feature1', 'feature2', 'correlation'])
        corr_df.to_csv(os.path.join(self.output_dir, 'feature_correlations.csv'), index=False)
        logger.info(f"Saved detailed correlation analysis to feature_correlations.csv")

    # Return all preprocessing results
    return {
        'scaled_data': scaled_data,
        'selected_data': selected_data,
        'feature_names': feature_names,
        'selected_feature_names': selected_feature_names,
```

```
        'scaler': scaler,
        'high_correlation_pairs': high_corr_pairs
    }
```

Appendix C: PostgreSQL Entiry Relationship Diagram



Appendix D: Python Method - Aggregated data query

```
comprehensive_query = text("""
            WITH project_metrics AS (
                -- Calculate average resource usage per project
                SELECT
                    p.id as project_id,
                    p.name as project_name,
                    p.staging_server_id,
                    p.production_server_id,
                    p.active,
                    AVG(sm.load_1) as avg_project_cpu_load,
                    AVG((sm.mem_total_kb - sm.mem_available_kb)::float / sm.mem_total_kb * 100) as
avg_project_memory_usage
                FROM wcmc_project_manager_project p
                LEFT JOIN wcmc_project_manager_servermetrics sm ON
                    (p.staging_server_id = sm.server_id OR p.production_server_id = sm.server_id)
                GROUP BY p.id, p.name, p.staging_server_id, p.production_server_id, p.active
            ),
            server_metrics AS (
                SELECT DISTINCT ON (ls.id)
                    ls.id as server_id,
                    ls.label as server_name,
                    ls.region,
                    sm.cpu_cores,
                    sm.mem_total_kb / 1024.0 as memory_mb,
                    sm.load_1 as cpu_load,
                    (sm.mem_total_kb - sm.mem_available_kb)::float / sm.mem_total_kb * 100 as memory_usage,
                    COUNT(DISTINCT p.id) as project_count,
                    COUNT(DISTINCT CASE WHEN p.active THEN p.id ELSE NULL END) as active_projects,
                    COUNT(DISTINCT CASE WHEN p.staging_server_id = ls.id THEN p.id END) as staging_projects,
                    COUNT(DISTINCT CASE WHEN p.production_server_id = ls.id THEN p.id END) as
production_projects,
                    AVG(pm.avg_project_cpu_load) as avg_project_cpu_load,
                    AVG(pm.avg_project_memory_usage) as avg_project_memory_usage,
                    string_agg(DISTINCT p.name, ', ' ORDER BY p.name) as project_names,
                    AVG(ii.amount) as avg_daily_cost
                FROM wcmc_project_manager_linodeserver ls
```

```
            LEFT JOIN wcmc_project_manager_servermetrics sm ON sm.server_id = ls.id
            LEFT JOIN wcmc_project_manager_project p ON (p.staging_server_id = ls.id OR
p.production_server_id = ls.id)
            LEFT JOIN project_metrics pm ON (pm.staging_server_id = ls.id OR pm.production_server_id =
ls.id)
            LEFT JOIN wcmc_project_manager_invoiceitem ii ON ii.linode_server_id = ls.id
            WHERE ls.active = true
            GROUP BY ls.id, ls.label, ls.region, sm.cpu_cores, sm.mem_total_kb, sm.load_1,
                sm.mem_available_kb
            ORDER BY ls.id, sm.cpu_cores DESC NULLS LAST
        )
        SELECT
            server_id,
            server_name,
            region,
            COALESCE(cpu_cores, 0) as cpu_cores,
            ROUND((COALESCE(memory_mb, 0) / 1024.0)::numeric, 1) as memory_gb,
            ROUND(cpu_load::numeric, 2) as cpu_utilization,
            ROUND(memory_usage::numeric, 1) as memory_utilization,
            project_count,
            active_projects,
            staging_projects,
            production_projects,
            ROUND(avg_project_cpu_load::numeric, 2) as avg_project_cpu_load,
            ROUND(avg_project_memory_usage::numeric, 1) as avg_project_memory_usage,
            project_names,
            ROUND(avg_daily_cost::numeric, 2) as avg_daily_cost,
            ROUND((avg_daily_cost * 30)::numeric, 2) as estimated_monthly_cost,
            ROUND((avg_daily_cost * 365)::numeric, 2) as estimated_yearly_cost,
            ROUND((cpu_load / NULLIF(cpu_cores, 0))::numeric * 100, 2) as cpu_per_core,
            ROUND((project_count / NULLIF(cpu_cores, 0))::numeric, 2) as projects_per_core,
            ROUND((avg_daily_cost / NULLIF(project_count, 0))::numeric, 2) as cost_per_project,
            ROUND((avg_daily_cost / NULLIF(cpu_cores, 0))::numeric, 2) as cost_per_core,
            ROUND((avg_daily_cost / NULLIF(memory_mb, 0))::numeric * 1024, 2) as cost_per_gb,
            ROUND((memory_usage / 100.0 * memory_mb / 1024.0)::numeric, 2) as memory_used_gb,
            ROUND((project_count / NULLIF(memory_mb / 1024.0, 0))::numeric, 2) as projects_per_gb
        FROM server_metrics
```

```
        """)
```

Appendix E: Python Method – Pre-process data

```
def preprocess_data(self, df: pd.DataFrame,
                    scaler_type: str = 'robust',
                    feature_selection: str = 'kbest',
                    correlation_threshold: float = 0.8) -> Dict:
    """
    Simplified preprocessing pipeline focusing on feature selection and scaling.

    Args:
        df: DataFrame containing the server metrics
        scaler_type: Type of scaler to use ('standard', 'robust')
        feature_selection: Feature selection method ('variance', 'kbest', None)
        correlation_threshold: Threshold for removing highly correlated features

    Returns:
        Dictionary with preprocessing results
    """
    logger.info("Preprocessing data...")

    # 1. Extract numeric features and handle missing values
    numeric_df = df.select_dtypes(include=['number'])
    numeric_df = numeric_df.fillna(numeric_df.median())

    # Store feature names
    feature_names = numeric_df.columns.tolist()
    logger.info(f"Working with {len(feature_names)} numeric features")

    # 2. Remove highly correlated features
    corr_matrix = numeric_df.corr()
    high_corr_pairs = []
    features_to_remove = set()

    for i in range(len(feature_names)):
```

```
            for j in range(i+1, len(feature_names)):
                if abs(corr_matrix.iloc[i, j]) > correlation_threshold:
                    high_corr_pairs.append((
                        feature_names[i],
                        feature_names[j],
                        corr_matrix.iloc[i, j]
                    ))
                    # Keep the feature with higher variance
                    var_i = numeric_df[feature_names[i]].var()
                    var_j = numeric_df[feature_names[j]].var()
                    if var_i < var_j:
                        features_to_remove.add(feature_names[i])
                    else:
                        features_to_remove.add(feature_names[j])

        # Remove highly correlated features
        features_to_keep = [f for f in feature_names if f not in features_to_remove]
        numeric_df = numeric_df[features_to_keep]
        feature_names = features_to_keep

        if high_corr_pairs:
            logger.info(f"Removed {len(features_to_remove)} highly correlated features")
            for f1, f2, corr in high_corr_pairs[:5]:  # Show only top 5
                logger.info(f"  {f1} and {f2}: {corr:.3f}")

            # Save correlation analysis to CSV
            corr_df = pd.DataFrame(high_corr_pairs, columns=['feature1', 'feature2', 'correlation'])
            corr_df.to_csv(os.path.join(self.output_dir, 'correlated_features.csv'), index=False)
            logger.info(f"Saved correlated features analysis to {os.path.join(self.output_dir,
'correlated_features.csv')}")

        # 3. Scale the data
        if scaler_type == 'standard':
            scaler = StandardScaler()
        elif scaler_type == 'robust':
            scaler = RobustScaler()
        else:
```

```
        logger.warning(f"Unknown scaler type: {scaler_type}. Using RobustScaler.")
        scaler = RobustScaler()

    scaled_data = scaler.fit_transform(numeric_df)

    # 4. Feature selection
    selected_data = scaled_data
    selected_feature_names = feature_names

    if feature_selection:
        if feature_selection == 'variance':
            # Remove low-variance features
            selector = VarianceThreshold(threshold=0.01)
            selected_data = selector.fit_transform(scaled_data)
            mask = selector.get_support()
            selected_feature_names = [feature_names[i] for i in range(len(feature_names)) if mask[i]]

            logger.info(f"Selected {len(selected_feature_names)} features using variance threshold")

        elif feature_selection == 'kbest':
            # First cluster the data to get pseudo-labels
            kmeans = KMeans(n_clusters=min(5, len(df) - 1), random_state=42, n_init=10)
            pseudo_labels = kmeans.fit_predict(scaled_data)

            # Select k best features based on ANOVA F-statistic
            k = max(int(len(feature_names) * 0.7), 5)  # Select 70% of features or at least 5
            selector = SelectKBest(f_classif, k=k)
            selected_data = selector.fit_transform(scaled_data, pseudo_labels)
            mask = selector.get_support()
            selected_feature_names = [feature_names[i] for i in range(len(feature_names)) if mask[i]]

            # Get feature scores
            scores = selector.scores_
            feature_scores = list(zip(feature_names, scores))
            sorted_features = sorted(feature_scores, key=lambda x: x[1], reverse=True)

            # Save feature selection results to CSV
```

```
            feature_selection_df = pd.DataFrame(
                sorted_features,
                columns=['feature', 'f_score']
            )
            feature_selection_df.to_csv(os.path.join(self.output_dir, 'feature_selection_results.csv'), index=False)
            logger.info(f"Saved feature selection results to {os.path.join(self.output_dir,
'feature_selection_results.csv')}")

        # Return all preprocessing results
        return {
            'scaled_data': scaled_data,
            'selected_data': selected_data,
            'feature_names': feature_names,
            'selected_feature_names': selected_feature_names,
            'scaler': scaler,
            'high_correlation_pairs': high_corr_pairs,
            'removed_features': list(features_to_remove)
        }
```

Appendix F: Python Method - Consensus Clustering

```
def _consensus_clustering(self, X: np.ndarray, all_results: Dict, method: str = 'majority_voting') -> Dict:
    """
    Perform consensus clustering using multiple methods and different numbers of clusters.
    """
    logger.info("="*50)
    logger.info("Starting Consensus Clustering Process")
    logger.info("="*50)

    # Extract cluster labels from individual algorithms
    logger.info("\nExtracting cluster labels from individual algorithms:")
    cluster_labels = {}
    for algo_name, result in all_results.items():
        n_clusters = result['n_clusters']
        logger.info(f"  - {algo_name}: {n_clusters} clusters")
        cluster_labels[algo_name] = result['cluster_labels']

    # Group results by number of clusters
    n_clusters_groups = {}
    for algo_name, labels in cluster_labels.items():
        n_clusters = len(np.unique(labels))
        if n_clusters not in n_clusters_groups:
            n_clusters_groups[n_clusters] = {}
        n_clusters_groups[n_clusters][algo_name] = labels

    # Calculate consensus for each number of clusters
    consensus_results = {}
    for n_clusters, group_labels in n_clusters_groups.items():
        logger.info(f"\nCalculating consensus for {n_clusters} clusters...")

        if method == 'majority_voting':
            # Convert labels to one-hot encoding
            one_hot_matrices = []
            for algo_name, labels in group_labels.items():
                one_hot = np.zeros((len(X), n_clusters))
                one_hot[np.arange(len(X)), labels] = 1
                one_hot_matrices.append(one_hot)
```

```
        # Average the one-hot matrices
        consensus_matrix = np.mean(one_hot_matrices, axis=0)

        # Perform majority voting
        consensus_labels = np.argmax(consensus_matrix, axis=1)

        # Calculate validation metrics
        sil_score = silhouette_score(X, consensus_labels)
        ch_score = calinski_harabasz_score(X, consensus_labels)
        db_score = davies_bouldin_score(X, consensus_labels)

        consensus_results[n_clusters] = {
            'cluster_labels': consensus_labels,
            'silhouette_score': sil_score,
            'calinski_harabasz_score': ch_score,
            'davies_bouldin_score': db_score,
            'consensus_matrix': consensus_matrix,
            'n_clusters': n_clusters,
            'algorithms_used': list(group_labels.keys())
        }

        logger.info(f"  Silhouette score: {sil_score:.4f}")
        logger.info(f"  Calinski-Harabasz score: {ch_score:.1f}")
        logger.info(f"  Davies-Bouldin score: {db_score:.4f}")

# Select the best number of clusters based on silhouette score
best_n_clusters = max(consensus_results.items(),
                      key=lambda x: x[1]['silhouette_score'])[0]
best_result = consensus_results[best_n_clusters]

logger.info("\nBest clustering configuration:")
logger.info(f"Number of clusters: {best_n_clusters}")
logger.info(f"Silhouette score: {best_result['silhouette_score']:.4f}")
logger.info(f"Calinski-Harabasz score: {best_result['calinski_harabasz_score']:.1f}")
logger.info(f"Davies-Bouldin score: {best_result['davies_bouldin_score']:.4f}")
logger.info(f"Algorithms used: {', '.join(best_result['algorithms_used'])}")
```

```
    # Plot cluster number selection
    plt.figure(figsize=(10, 6))
    n_clusters_list = sorted(consensus_results.keys())
    sil_scores = [consensus_results[n]['silhouette_score'] for n in n_clusters_list]
    ch_scores = [consensus_results[n]['calinski_harabasz_score'] for n in n_clusters_list]
    db_scores = [consensus_results[n]['davies_bouldin_score'] for n in n_clusters_list]

    plt.plot(n_clusters_list, sil_scores, 'o-', label='Silhouette Score')
    plt.plot(n_clusters_list, ch_scores, 's-', label='Calinski-Harabasz Score')
    plt.plot(n_clusters_list, db_scores, '^-', label='Davies-Bouldin Score')

    plt.axvline(x=best_n_clusters, color='r', linestyle='--', label=f'Best: {best_n_clusters} clusters')
    plt.xlabel('Number of Clusters')
    plt.ylabel('Score')
    plt.title('Cluster Number Selection')
    plt.legend()
    plt.grid(True, alpha=0.3)
    plt.tight_layout()
    plt.savefig(os.path.join(self.output_dir, 'cluster_number_selection.png'))
    plt.close()

    # Add individual results and consensus results to the best result
    best_result['individual_results'] = all_results
    best_result['consensus_results'] = consensus_results

    return best_result
```

Appendix G: Aggregated Data with Clustering

server_id	cluster	cpu_utilization	memory_utilization	project_count	staging_projects	production_projects	cost_per_project	avg_project_cpu_load	avg_project_memory_usage	avg_daily_cost
860	0	0.75	22.3	3	3	0	2.63	0.81	21.8	7.9
878	0	0.55	15.3	10	10	0	1.96	0.52	16.2	19.58
850	0	0.08	34.6	2	2	0	7.9	0.05	23.5	15.8
852	0	0.03	12.4	4	0	4	7.79	0.04	18	31.16
851	0	0	19.4	2	2	0	3.96	0	18.2	7.93
856	0	0.51	18.1	0	0	0	0	0	0	7.9
857	0	0.41	25.8	0	0	0	0	0	0	15.86
858	0	0.1	19.2	0	0	0	0	0	0	15.8
859	0	2.03	63.7	0	0	0	0	0	0	15.8
861	0	0.42	12.8	0	0	0	0	0	0	31.6
862	0	0.61	21.7	0	0	0	0	0	0	15.8
863	0	0	10.4	0	0	0	0	0	0	7.89
864	0	0.03	21.6	0	0	0	0	0	0	10.71
865	0			0	0	0	0	0	0	3.95
867	0	0	52.5	0	0	0	0	0	0	7.93
868	0	0	17	0	0	0	0	0	0	3.95
869	0	0	7.6	0	0	0	0	0	0	15.86
870	0	0.16	6.4	0	0	0	0	0	0	15.86
871	0	0	19.9	0	0	0	0	0	0	15.86
872	0	0.19	27.6	0	0	0	0	0	0	8.56
875	0	0	14.1	0	0	0	0	0	0	10.6
876	0	0	70.1	0	0	0	0	0	0	31.26
879	0	0.13	31	0	0	0	0	0	0	30.79
880	0	0.47	8.4	0	0	0	0	0	0	19.17
882	0	0.1	7.4	0	0	0	0	0	0	58.73
883	0	0.08	4.4	0	0	0	0	0	0	27.2
885	0	0.27	36.6	0	0	0	0	0	0	6.35
887	0	0	23.4	0	0	0	0	0	0	3.81
888	0	0.08	21.1	0	0	0	0	0	0	3.81
889	0	0	0	0	0	0	0	0	0	3.6
890	0	0	0	0	0	0	0	0	0	2.97
891	0	0.35	20	0	0	0	0	0	0	22.64
893	0	0	0	0	0	0	0	0	0	3.56
855	1	0.64	35.7	2	1	1	1.98	0.64	35.7	3.95
892	1	0.37	30.6	4	0	4	5.61	0.38	33	22.46
866	1	0.29	36.9	2	2	0	1.98	0.29	36.9	3.95
886	1	0	32.1	3	0	3	2.12	0.24	32.1	6.35
877	2	0.37	16.8	1	1	0	21.34	0.37	16.8	21.34
881	2	0.02	6.6	2	2	0	14.68	0.02	6.6	29.36
854	2	0.02	14.5	1	0	1	15.69	0.01	17	15.69
874	3	1.65	29.8	36	36	0	0.34	1.65	29.8	12.4
853	4	0.66	41.8	15	15	0	0.37	0.68	42.3	5.55
884	4	0.43	40.5	10	10	0	2.25	0.38	38.6	22.54
873	5	0	15.3	1	1	0	31.14	0	15.3	31.14