UNIVERSITY OF CALIFORNIA

Los Angeles

Generative Data Science: Applications for Early Life Cycle Cost Estimation in the Aerospace Industry

A thesis submitted in partial satisfaction
of the requirements for the degree
Master of Science in Statistics

by

Trevor Ryan Beer

2024

ABSTRACT OF THE THESIS

Generative Data Science: Applications for Early Life Cycle Cost Estimation in the Aerospace Industry

by

Trevor Ryan Beer
Master of Science in Statistics
University of California, Los Angeles, 2024
Professor Guang Cheng, Chair

Cost estimation in the early mission life cycle is typically a difficult task, with the standard being utilizing historical analogues and heavy reliance on systems engineers with exceptional domain expertise to guide analysis. However, the lack of data points (previous flown missions) introduces large amounts of uncertainty and can call into question the validity of statistical results. Nonetheless, this task is of utmost importance in the context of exploring mission feasibility and particularly when trying to solicit work for a NASA Announcement of Opportunity. This paper explores the utilization of generative data science in the low-data paradigm, in order to train more effective and robust cost models. The development of generative methods and models to protect privacy and assist various machine learning tasks has been given much theoretical focus, but these frameworks have not been adopted widely in many realms that present great opportunities. Furthermore, we compare different generative methods by assessing the quality of marginal distributions, maintenance of conditional relationships that are grounded in physical spacecraft parameters, and by exploring their utility in simple machine learning tasks that represent the types of analyses done in mission formulation.

To all those who provided support, without which I would not be writing this

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

I would like to express my utmost gratitude to those who provided guidance and mentoring that was instrumental to the completion and quality of this project. First, I would like to sincerely thank Professor Guang Cheng and the rest of my thesis committee, who provided invaluable guidance and feedback in taking this thesis from concept to reality.

Additionally, Dr. Chi-Hua Wang played a critical role in helping me develop and articulate the goals of this project clearly. His insight, subject matter expertise, and overall mentorship were vital to the success of my research.

Furthermore, I am forever grateful for the conversations and presentations I had the opportunity to be a part of in the Trustworthy AI Lab at UCLA. The various perspectives and flow of ideas from all the bright minds I was fortunate enough to call colleagues helped my problem solving approach and undoubtedly supplemented my ability to effectively conduct my research.

Additionally, I would like to thank my colleagues at the Jet Propulsion Laboratory (JPL), who helped inspire my industry application and identify this opportunity to integrate my education with real world data.

I am truly fortunate to have such an accomplished and talented cohort of peers and mentors during my graduate education, and express my deepest appreciation to them all.

CHAPTER 1

Introduction

Machine learning refers to the widely popular methodology of applying statistically and mathematically driven algorithms to data with the goal of learning trends and generalizing to unobserved data. Generative data science is a booming subset of machine learning that is focused upon learning the structure and patterns present in the data it is given to generate new data that maintains the learned properties from the original set. Among motivating factors to employ generative methods on a given dataset are preserving the privacy when a dataset contains unique personal identifiers, bolstering the size of data for deep learning and data intensive learning tasks, and creating more fair data by addressing class imbalances. With the surge of generative models in the public eye, such as ChatGPT (OpenAI, 2024), the adoption and evaluation of these models on industry specific tasks has become a high priority research task.

When working with small datasets, machine learning and particularly generation remains a complex and paradoxical task in nature. In these cases, the individual or party working with data would seek to augment the dataset in order to improve predictive power and results for downstream tasks, however the training data may not sufficiently cover the distribution of the real data well. This problem harms the utility of generated data, creating a cycle in which one can neither reliably conduct traditional machine learning analysis, nor can they reliably create samples that generalize to unseen entries well. Considering only the subset of data that is tabular in form, this phenomenon becomes more pronounced as tabular data cannot be transformed or augmented as is done to image data or other relevant formats. Understanding the balance between fidelity and utility in such scenarios and how to make effective decisions that permit the best performance of generated data from a statistical and industry specific perspective is essential and perhaps of more importance in this setting, relative to the typical large data situations most generally considered in the generative data science literature.

Cost estimation during early space mission formulation is an essential part of the process of determining mission feasibility in the aerospace industry. Generating good estimates is imperative, seeing as underestimation lends a mission at risk to suffer schedule delays and budget overruns, while overestimation can lead to rejection of an otherwise sound proposal or inhibit the funding of additional science and research ventures if selected. Thus, it is of great interest to utilize reliable data-driven methodology and machine learning methods to predict cost using various parameters from previously flown missions. However, due to the infrequent nature of major missions/launches, the available datasets are small in nature. Therefore, systems engineers often find themselves restricted to basic machine learning methodology (linear regression, KNN) or even non-parametric analysis by analogy to a similar mission. Even when parametric analysis is viable, uncertainty in parameter estimates or the lack of ability to effectively use a train/test split of the data makes model selection and validation a cumbersome task. Therefore, it follows that cost engineers would take interest in reliable data augmentation methods to assist with scenario exploration and the availability of real data for model evaluation.

This paper seeks to build on the corpus of work done in both generative data science and cost estimation in the aerospace industry by applying different generative methods to a dataset that reflects the lack of specific design information and uses very general parameters to characterize a mission as in the early life cycle. The generated data will be evaluated against the original dataset to ensure the fidelity of the synthetic samples. Additionally, simple parametric analyses will be leveraged to make conclusions about the utility of the generated data. Though privacy is not a goal or focus of this use case, it is an integral aspect of synthetic data evaluation in many contexts and thus brief discussions of how to incorporate and judge the privacy preserving qualities will be given. Ultimately, the goal of this thesis is to demonstrate that the advances in generative data science equip cost and systems engineers with more tools than ever before to combat issues that arise in small datasets, without sacrificing the integrity of the data. The presented methodology is intended to be statistically rigorous, intuitively justifiable, and perhaps most importantly simple to implement.

CHAPTER 2

Generative Data Science

In this section, we present a sweeping overview of the past and present state of generative data science with an eye towards applications on tabular data specifically. The core concepts and methods that have driven innovation in this field will be fleshed out, with an eye towards specific applications on tabular data and the advantages or disadvantages present in different frameworks.

2.1 Tabular Data

We focus on the tabular data setting in this paper and analysis. Tabular data can be formally presented in the following manner following (Zhang et al., 2017): we consider a dataset D of size n (rows). This dataset has some set of attributes $\mathcal{A}$ of size d which constitutes the columns. Every $X_i \in \mathcal{A}$ is either discrete or continuous. As such, we can consider each column to be a random variable allowing us to consider the respective marginal distributions $p(X_i)$, as well as the overall joint distribution, $p(X_1, \ldots, X_d)$. In the supervised machine learning setting we consider for downstream evaluation later in this thesis, we consider one of the columns to be a target variable Y we seek to estimate. Thus, the notation $p(X, Y)$ may be used to refer to the joint distribution of all the data.

When discussing tabular data in the generative context, it has a number of notable drawbacks relative to other forms of data, such as images. For instance, data augmentation can be conducted with images by geometrically transforming the image (scaling, cropping, rotating) or scaling pixel values to alter qualities like brightness or contrast. These augmentation methods simply do not exist for tabular data, incentivizing a synthetic data approach to enriching datasets.

2.2 History and Overview

The idea of generating synthetic data is certainly not a novel one, as physics informed simulations have been employed in the sciences long before statistically driven methods entered the public sphere. While not entirely relevant to the methods we will discuss below, these simulations and scientific modeling in general were instrumental predecessors to the work done over recent decades in synthetic data generation, and demonstrate the fact that statistically based generation represents a new approach to a set of problems that are not novel themselves.

Moving to a statistical perspective, researchers began to explore the ability to use data imputation and perturbation as a means of generating data that could be disseminated to the public realm for analysis while preserving the privacy of parties with personally identifiable information (PII) (Rubin, 1993). Rubin and others early work emphasized the privacy as the impetus to drive research in the field. Truly though, it is the last decade in which generating synthetic data has blossomed and competing models have emerged, all with different advantages and disadvantages. We will present a broad overview of some major models and architectures that are prevalent in the field today here, with a more detailed look to come in the subsequent sections regarding models of interest in the scope of this thesis.

The simplest and perhaps most intuitive starting point for a generation scheme is that of using marginal distributions. Effectively these methods seek to sample from the marginal distributions of columns in the tabular data. In doing so in an iterative manner for each column, an entirely new dataset of synthetic samples can be built. Despite the apparent simplicity of marginal-based methods, they remain exceptionally relevant and useful in practice, especially when constraints such as data size rule out the possibility of more complex methodology. The MST algorithm (McKenna et al., 2021) is one prominent example of this school of thought, as it samples from low-dimensional marginal distributions that have had noise added to them to preserve privacy, providing a general and flexible framework that scales well to different datasets. The independent histogram mode featured in DataSynthesizer (Ping et al., 2017) represents another classic approach that assumes feature independence and uses estimated marginal distributions to build synthetic samples.

Bayesian networks are another popular schema for generation, which seek to model relationships between variables in a graphical manner. In treating each variable X_i as a node of the graph, the edges between nodes can then be interpreted as the relationship between variables in a dataset, which can be visually expressed as a directed acyclic graph (DAG). The generation process is similar in concept to the marginal methods described above, as one samples from the joint distribution of variables in the graph to create new synthetic samples. PrivBayes (Zhang et al., 2017) is among the most ubiquitous of the Bayesian graphical models, adopting a GreedyBayes algorithm to learn a Bayesian network on the data, approximating a joint distribution from the conditional distributions specified by the network with noise added to preserve privacy, and finally sampling from the approximated distribution.

The field of computer science has also lent significant contributions to generative data science, especially when considering approaching synthetic data generation from a deep learning perspective. Generative adversarial networks (GANs) emerged in the 2010s with incredible results on image, language, and translation tasks (Goodfellow et al., 2014). The major concept behind a GAN is that of a zero-sum game. The architecture has two neural networks, one aptly named the "generator" $\mathcal{G}$ and the second being the "discriminator" $\mathcal{D}$. These two networks are in constant competition with one another, with the generator being tasked with fooling the discriminator, and the discriminator aiming to distinguish a generated sample from the real data. This process leads to simultaneous training of each network, and the mathematical expression of this game is as follows:

$$\min_{\mathcal{D}} \max_{\mathcal{G}} V(\mathcal{D}, \mathcal{G}) = \mathbb{E}_{x \sim p_{data}(x)}[\log \mathcal{D}(x)] + \mathbb{E}_{z \sim p_z(z)}[\log(1 - \mathcal{D}(\mathcal{G}(z))]$$

Here x is the real sample and z is a noise vector, with p denoting the respective distributions of each. The training process is very standard for a deep learning network, as after the generator has created a sample and it has been fed to the discriminator for classification, errors are back-propagated through the network, updating both $\mathcal{G}$ and $\mathcal{D}$ until some loss objective is met, from which point the generator is used to create synthetic samples that aim to be indistinguishable from real data. However, training a GAN is not a simple task in practice. One of the most common failure modes in GAN training is mode collapse, in which the generator learns one (or a small set of) output that fools the discriminator every time (Zhang et al., 2018). Thus, the generator effectively collapses and cannot produce diverse samples. Generally, this and other issues arise due to the complex nature of simultaneously training each network, as there is always a risk that the discriminator will learn faster than the generator, or vice versa.

As stated, the GAN architecture originally rose to prominence for its results on image datasets, however recent advances such as the advent of CTGAN (Xu et al., 2019) and related methods have made this methodology viable for tabular data. Without modification, a vanilla GAN will struggle with most types of tabular data as it does not handle the presence of categorical features well. However, CTGAN combats this using an embedding scheme that allows categorical features to be mapped to continuous space. Thus, the issue of categorical features is bypassed, and CTGAN also features conditional sampling to help combat class imbalance issues with training data. CTGAN has been proven to not be a differentially private generation mechanism, leading researchers to propose additional algorithms such as DPGAN (Xie et al., 2018) and PATE-GAN (Jordon et al., 2018) to address use cases in which PII or other factors dictate that privacy is a major concern.

Variational Autoencoders (VAEs) are another popular generative method, with TVAE (Xu et al., 2019) standing out as a classical state of the art algorithm example in the field. VAEs, like GANs, have two network components, in this architecture called an encoder and decoder. The encoder will take the input data and project it to some latent space, creating a distribution $q_\phi(z|x)$. A sample is drawn from this latent distribution, and decoded, at which point error is back-propagated through the network as in traditional deep machine learning methods. Formally, the loss term can be expressed as:

$$\text{Loss} = \mathbb{E}_{p_{data}}(x)[\log p(x) - D_{KL}(q_\phi(z|x)|p_\theta(x|z)]$$

Here $p_\theta(x|z)$ is the true posterior distribution and D_{KL} is the Kullback-Leibler divergence. The loss term is aiming to optimize maximize the log-likelihood term while regularizing for the distance between the latent distribution and the true posterior, which are parametrized by ϕ and θ respectively. VAEs in general are considered easier to train than GANs and can produce similar results, however in image generation GANs tend to have a noteworthy increase in sharpness.

Much like how diffusion models have grown to surpass GAN/VAE architectures with image and related data, diffusion models have been modified to handle the tabular setting while showing very promising results when benchmarked against GAN/VAE architectures. Diffusion models, more formally denoising diffusion probabilistic models (DDPM), take inspiration from the thermodynamic concept of diffusion. These models consist of three major components: the forward (noising) process, the reverse (denoising) process, and the sampling procedure. During the forward process, Gaussian noise is added for some series of timesteps until the original distribution it itself Gaussian. The forward process is often a Markov process, allowing for the sequential expression of the data distribution during the noising process as follows: given a starting distribution $x \sim q(x_0)$:

$$q(x_1, \ldots, x_T|x_0) = \prod_{t=1}^{T} q(x_t|x_{t-1})$$

The denoising process can be formally expressed in a similar manner:

$$p_\theta(x_{t-1}|x_t) = N(x_{t-1}; \mu_\theta(x_t, t), \Sigma_\theta(x_t, t))$$

TabDDPM (Kotelnikov et al., 2023) is a popular implementation of the diffusion methodology to tabular data that has shown particularly good results with regards to privacy guarantees.

Returning to the context of the low-data paradigm we introduced in the first chapter, it is incredibly likely that the above methods, despite being state of the art in most cases, will not be successful. While there is not a formal lower bound with regards to a data requirement to train any of these models, many common problems when working with low-data such as overfitting, class-imbalance, and generalization issues will be accentuated when employing complex architectures such as those described above. As such, we included the above paragraphs to create a more complete image of the generative data science landscape, but further elaboration on the above methods will be omitted for brevity as they are not relevant to the final analyses and comparisons we conduct.

2.2.1 Differential Privacy

While the use case we present in this paper does not consider privacy a major concern, it is impossible to discuss the field of generative data science without introducing the notion of differential privacy and motivation behind it. While pioneers in the field such as Rubin had considered the privacy guarantees of generated data, the work of Cynthia Dwork is that which first formally presented the gold standard of privacy assessment in generative data science: differential privacy (Dwork and Roth, 2014). The rigorous study of differential privacy hinges on the intersection between privacy and utility when working with generated data. The main idea behind differential privacy is that by injecting noise into the estimated distribution from which a sample is drawn, the information of any individual within the original set is better protected.

More formally, we consider two datasets D and D' which are identical except for one entry. This constitutes the definition of neighboring datasets. A synthetic dataset is differentially private if an adversary or attacker cannot learn any more information from one dataset than another, that is the presence or absence of any individual entry does not impact the final dataset (Dwork, 2008). A very standard presentation of differential privacy in literature is that of (ϵ, δ)-differential privacy. Mathematically the definition is generally constructed as:

$$\mathbb{P}(\mathcal{M}(D) \in S) \geq \exp(\epsilon)\mathbb{P}(\mathcal{M}(D') \in S) + \delta$$

Where $\mathcal{M}$ is a randomized algorithm, in our case the generator $\mathcal{G}$, and the statement holds $\forall S \in \mathcal{M}$. The ϵ parameter refers to the maximum distance between identical queries on D and D', while the δ represents the probability of information leakage. Another common practice in literature is to set $\delta = 0$ and consider ϵ-differential privacy:

$$\mathbb{P}(\mathcal{M}(D) \in S) \geq \exp(\epsilon)\mathbb{P}(\mathcal{M}(D') \in S)$$

Taking the concept to the extreme, as ϵ is taken to 0, our definition is further reduced to

$$\mathbb{P}(\mathcal{M}(D) \in S) \geq \mathbb{P}(\mathcal{M}(D') \in S)$$

which tells us that no individual row in any dataset provides additional information than the rest of the data already gives.

In practice, ϵ-differential privacy is often incorporated into generative models by injecting Laplacian noise at some stage between the input of real data and sampling of synthetic data. Setting the location parameter to 0 and the scale or sensitivity parameter to some function of ϵ, for example $\frac{1}{n\epsilon}$ as in DataSynthesizer permits explicit control over the degree to which differential privacy is injected. The Laplacian mechanism is common in practice due to the fact that it has well studied empirically researched guarantees within the presented ϵ-differential privacy framework.

When considering the incorporation of differential privacy in synthetic data generation, data size is an important consideration. For example, considering the extreme example of one dataset with 2 entries, and another with 1 billion entries, the concept at hand becomes glaringly apparent. Removing one entry from the first dataset essentially removes half the information available to make any statistical conclusions, while only removing a fraction of a percent of the information in the larger set. Clearly, this relates directly to the ϵ

parameter as well, as injecting the same amount of noise into datasets of different sizes can have wildly different outcomes with regards to downstream utility tasks, with which there is a trade-off. Researchers approach privacy related problems from a number of different attack scenarios and perspectives, which are outside of the scope of this paper and as such will be omitted.

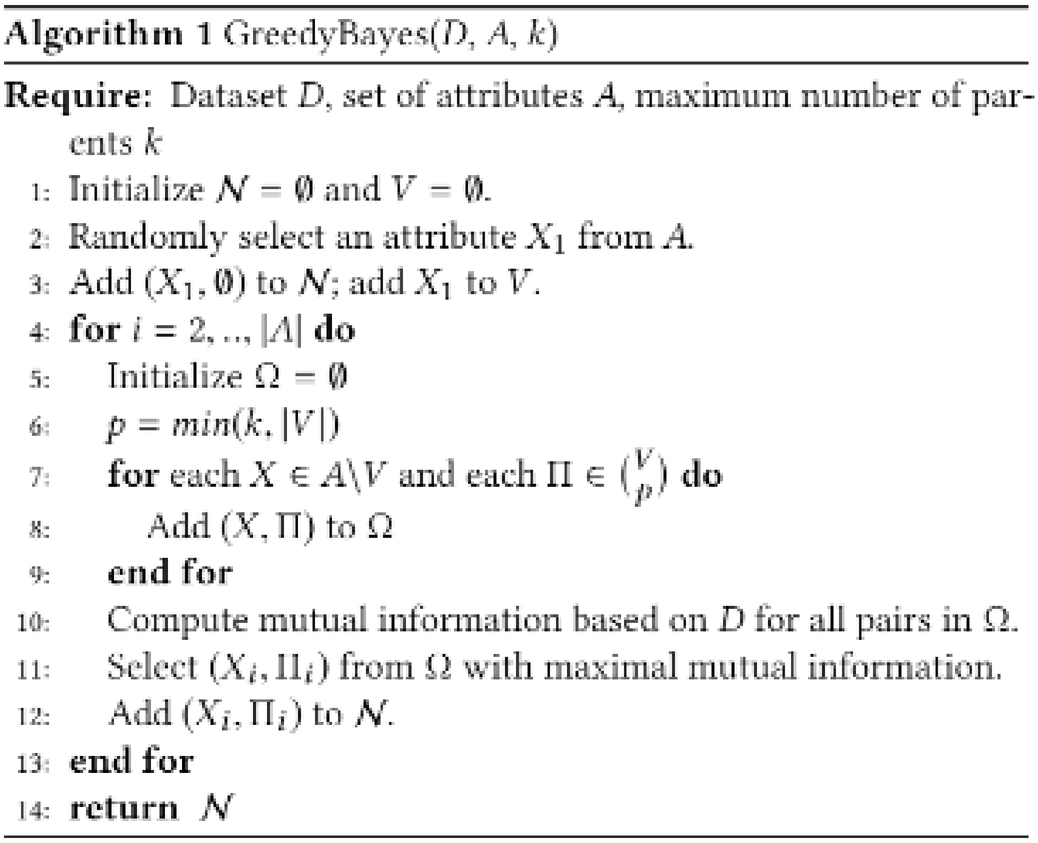
The following sections present a more complete image of a couple frameworks that we identified as having the most potential to perform well given the constraints presented by the data.

2.3 DataSynthesizer

DataSynthesizer (Ping et al., 2017) is a tool that builds heavily upon the marginal-based and Bayesian network based methodologies presented above, giving users a highly flexible tool that can be scaled in complexity based on the demands of the data that is input. The tool consists of three components: a DataDescriber, DataGenerator, and ModelInspector. Additionally, the tool supports three different implementation modes: random, independent attribute, and correlated attribute, which will be described in more detail below. The random mode simply treats the marginal distribution of each variable as uniform between the minimum and maximum values present in the data respectively, and samples from this uniform distribution. Due to the naive and impractical nature of implementing this on a complex dataset, further explanation or analysis considering this mode will be omitted.

When in the independent attribute mode, the DataDescriber module will infer the data type from four classes (string, integer, float, and datetime), and calculate the domains and estimated distributions of each feature. The estimated distributions are calculated as a frequency histogram, from which samples are drawn independently from each variable to construct new records.

Similarly, the DataDescriber in correlated attribute mode uses the GreedyBayes method outlined in Figure 2.1 in order to construct a Bayesian network and define the sampling order for drawing the conditional samples from. Essentially it samples a root attribute as defined by the network, and from there builds out the rest of the column values by sampling from the conditional distributions. The sampling algorithm employed by the DataGenerator as in the paper is presented in Figure 2.2.

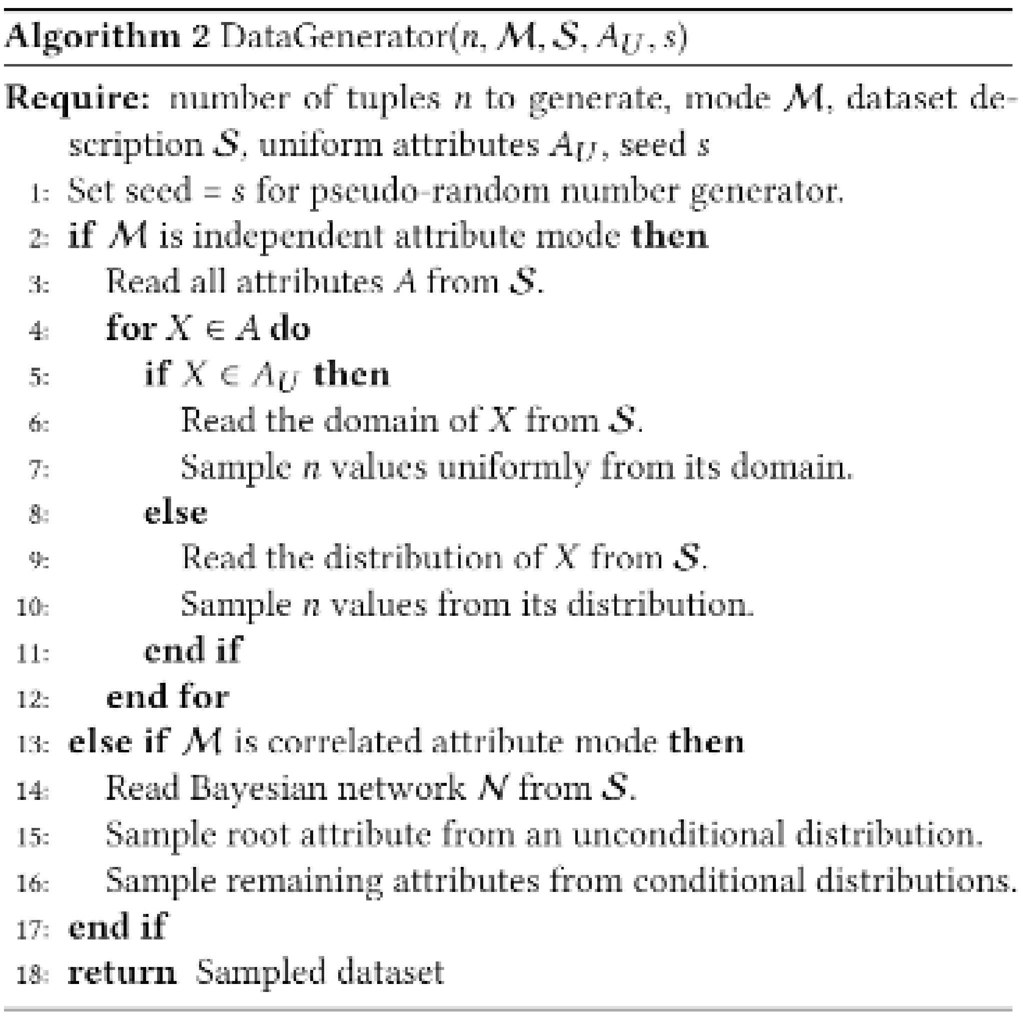


Algorithm 1 GreedyBayes(D, A, k)

```
Require: Dataset $D$, set of attributes $A$, maximum number of parents $k$
 1: Initialize $\mathcal{N} = \emptyset$ and $V = \emptyset$.
 2: Randomly select an attribute $X_1$ from $A$.
 3: Add $(X_1, \emptyset)$ to $\mathcal{N}$; add $X_1$ to $V$.
 4: for $i = 2, .., |A|$ do
 5:     Initialize $\Omega = \emptyset$
 6:     $p = min(k, |V|)$
 7:     for each $X \in A \setminus V$ and each $\Pi \in \binom{V}{p}$ do
 8:         Add $(X, \Pi)$ to $\Omega$
 9:     end for
10:     Compute mutual information based on $D$ for all pairs in $\Omega$.
11:     Select $(X_i, \Pi_i)$ from $\Omega$ with maximal mutual information.
12:     Add $(X_i, \Pi_i)$ to $\mathcal{N}$.
13: end for
14: return $\mathcal{N}$
```

Figure 2.1: GreedyBayes Algorithm as presented in DataSynthesizer

All implementation modes allow for control over an ϵ parameter that controls the amount of noise injected into the sampling distributions with the theoretical guarantee of differential privacy discussed above. However, considering our use case in which the data is to be handled by industry experts to augment existing data rather than to be shared outside of a company or agency's ecosystem, this parameter will be toggled off and differential privacy will not be considered.



Algorithm 2 DataGenerator(n, $\mathcal{M}$, $\mathcal{S}$, A_U, s)

```
Require: number of tuples $n$ to generate, mode $\mathcal{M}$, dataset description $\mathcal{S}$, uniform attributes $A_U$, seed $s$
 1: Set seed = $s$ for pseudo-random number generator.
 2: if $\mathcal{M}$ is independent attribute mode then
 3:     Read all attributes $A$ from $\mathcal{S}$.
 4:     for $X \in A$ do
 5:         if $X \in A_U$ then
 6:             Read the domain of $X$ from $\mathcal{S}$.
 7:             Sample $n$ values uniformly from its domain.
 8:         else
 9:             Read the distribution of $X$ from $\mathcal{S}$.
10:             Sample $n$ values from its distribution.
11:         end if
12:     end for
13: else if $\mathcal{M}$ is correlated attribute mode then
14:     Read Bayesian network $\mathcal{N}$ from $\mathcal{S}$.
15:     Sample root attribute from an unconditional distribution.
16:     Sample remaining attributes from conditional distributions.
17: end if
18: return Sampled dataset
```

Figure 2.2: Generation Algorithm used by DataSynthesizer

2.4 Curated LLM

In contrast to the relatively simple, tried and true process used above with DataSynthesizer, we also consider the ability of large-language models (LLM's) to generate synthetic samples for this project, specifically a generation/curation schema set forth in CuratedLLM (Seedat et al., 2024).

Considering prior research in the literature, LLM's have become increasingly prevalent since ChatGPT made waves in late 2022. Originally lauded for their performance in natural language processing tasks and text generation, the possibility for LLM's to be developed came into being after the release of "Attention is All You Need" (Vaswani et al., 2023), which opened the door for transformer based architectures, such as the Google's Bidirectional Encoder Representations from Transformers (BERT), which achieves state of the art results on natural language processing tasks, though it does not have generative text abilities due to the lack of a decoder structure in the model (Devlin et al., 2019).

Before describing how this relates to the task of data generation, we must first build a brief understanding of how an LLM works. The transformer architecture incorporates an attention mechanism that allows for increased contextual understanding when given a sequence as input. The general attention mechanism is defined as follows:

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

where Q, K, V correspond to query, key, and value vectors respectively, while d_k corresponds to the dimensionality of the keys. Attention can take different forms, such as self-attention, in which Q, K, V all come from the same source vector, or cross-attention, in which the Q vector is derived from a different vector than K and V. This language is used because the input vector for a LLM is tokenized, or turned into a numerical vector representation of tokens in a dictionary that is the set of all possible tokens. The encoded vector is passed through both attention and feed-forward layers during model training, and eventually the encoder output is passed through a decoder network similarly comprised of attention and feed-forward neural network layers. ChatGPT is unique in that it utilizes a decoder-only transformer architecture. Architectural nuances such as the merits of encoder decoder vs decoder only models are outside the scope of the analysis, but worth mentioning as the rapid acceleration of LLM related research may make these distinctions more instrumental to understand as benchmarking and evaluation procedures improve and develop.

Recently more research has been devoted to exploring the capabilities of LLM's as tabular data generators. Generation of Realistic Tabular data (GReaT) identifies how to leverage the contextual understanding capabilities of transformer based LLM's by textually embedding the data and prompt engineering to produce synthetic data that compares to state of the art methods (Borisov et al., 2023). Additionally, LLM's have proven to reduce the burden of pre-processing that is common with other generative methods such as CTGAN. Data preparation presents the opportunity for information to be lost from the unaltered dataset, and steps such as one-hot encoding of categorical features or imputation (or removal) of missing data can undermine the utility of generated data before the generation has even been conducted.

The methodology presented in CuratedLLM builds upon the existing methods and motivations described above to generate data in an ”low data regime”, defined as $n < 100$ samples. Pre-trained LLM's are trained on an immense corpus of text data, and as such have the ability to contextualize and leverage what is learned without additional fine-tuning which is computationally expensive and perhaps not feasible in many cases. Therefore, using a frozen LLM, in this case ChatGPT4.0, will be the context in which further concepts related to CuratedLLM are discussed.

The key novel aspect in this methodology is the data curation step, in which the learning dynamics of the synthesized data are analyzed via predictive confidence and assessing the aleatoric uncertainty associated with each sample. This is accomplished by training a supervised model on the real training data, computing the quality metrics mentioned above, and using those findings to create the curated synthetic dataset. The motivation for curation is the fact that the LLM is estimating the joint distribution of the real data, and this generation is likely to have inherent noise and contain mislabeled samples. Therefore, in evaluating and discarding samples that have been identified as noisy, we may be left with a higher quality dataset that will have better downstream performance. A general visual overview of this method is presented in Figure 2.3.

Learning dynamics refers to the fact that individual samples of a dataset contain information about the nature of these samples, which permits experimental setup that can actively test sample quality and use it to judge the fidelity of generated data.

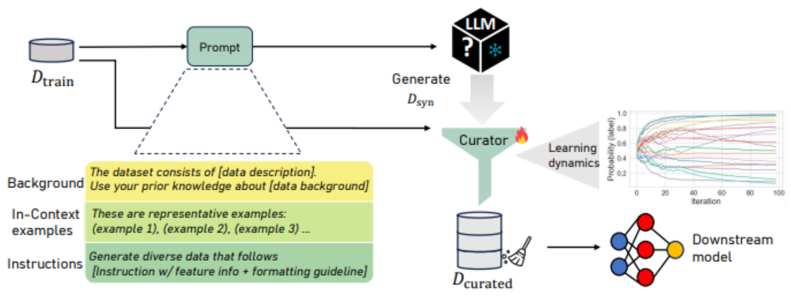

Prompt
LLM
D_{train}
Generate D_{syn}
Curator
Learning dynamics
Probability (label)
Iteration
Background
The dataset consists of [data description]. Use your prior knowledge about [data background]
In-Context examples
These are representative examples: (example 1), (example 2), (example 3) ...
Instructions
Generate diverse data that follows [Instruction w/ feature info + formatting guideline]
D_{curated}
Downstream model

Figure 2.3: Workflow outlined in CuratedLLM

The proposed curation mechanism follows this formal statistical setup: Train a classifier f on real data, D_{train}. During the training, the model should go through E checkpoints such that we have a set of models $\mathcal{F} = \{f_1, \ldots, f_E\}$ corresponding to the classifier at each checkpoint. Now, define H, a uniform random variable over the set of checkpoints $\mathcal{F}$. Then, we have the following two definitions for the conditional correctness of the model predictions for a given checkpoint (in the binary case):

$$\mathbb{P}(\hat{Y}_{\mathcal{F}}(x, y) = 1 | H = h) = [h(x)]_y$$

$$\mathbb{P}(\hat{Y}_{\mathcal{F}}(x, y) = 0 | H = h) = 1 - [h(x)]_y$$

This formal setup allows the further definition of both average confidence and aleatoric uncertainty of a sample, provided below:

$$\textbf{Average Confidence: } \frac{1}{E}\sum_{i=1}^{E}[f(x)]_y$$

$$\textbf{Aleatoric Uncertainty: } \frac{1}{E}\sum_{i=1}^{E}[f(x)]_y(1 - [f(x)]_y)$$

Once these values are determined, the sample curation is a result of comparing the confidence and uncertainty scores to some threshold τ_{conf} and $\tau_{aleatoric}$ respectively, and discarding samples for which the minimum threshold is not met.

In practice, it is recommended that an algorithm such as XGBoost (Chen and Guestrin,

2016) is utilized to train the curation model. This arises from the fact that the curation model should be at least as flexible as any desired downstream task, encouraging the adoption of an architecture that performs well from a flexibility and generalizability standpoint. The authors consider the binary classification case, and consider predicting the same variable for both the curation step and downstream tasks. We consider both regression and classification tasks, making the performance of curated data for a downstream task that is not inherently tied to the curation process an interesting question to be aware of during analysis.

This approach is incredibly promising, however we must also consider the drawbacks presented and how that may impact the ability to generate realistic and useful synthetic data. In the scope of this paper, we use publicly sourced data and ChatGPT4.0 to generate samples, however entire body of training data for a model like ChatGPT may include public data of various sources. This presents memorization risk and as such the sourcing of any data as well as pre-processing done should be considered before blind application of LLM-based analyses. Furthermore, company or agency policy may dictate that third-party AI tools cannot be used with sensitive data, restricting the ability to utilize these methods. As of recently the NASA was still developing an approach regarding the use of AI tools, taking the reasonable stance that generative AI tools are not to be used with sensitive data (Heilweil and Alder, 2023).

CHAPTER 3

Aerospace Industry Background and Motivation

In order to best analyze the fidelity and utility for our generated data in this thesis, a strong understanding of the mechanics of the early mission life cycle in the aerospace industry is necessary. This chapter intends to describe the role of cost and systems engineers at these early stages while providing discussion surrounding the current best practices and challenges associated with this job, and thus where the opportunity to leverage synthetic data presents itself.

3.1 Mission Formulation

As one may suppose, groundbreaking science missions do not occur overnight. Along the way from a simple idea or sketch on a napkin, to the expanses of space for mission operations, a concept much go through a rigorous development process, with the maturity benchmarked throughout this process. The use of a descriptive concept maturity level (CML) (Wessen et al., 2013) is one manner in which the development process is assessed. This framework considers 8 discrete steps that correspond to milestones with respect to implementation. Namely, it is the second and third steps, "Initial Feasibility" and "Trade Space", respectively, that cost and systems engineers conduct the most work. In fact, the earliest stages in formulation and proposal generation require extensive resources and significant institutional investment. Note that the CML framework is not necessarily linear, and that Figure 3.1 contains a flow chart that more accurately depicts the iterative and fluid nature of formulation activities (National Academies of Sciences, Engineering, and Medicine, 2022). The proposal process is competitive in this context as well, with no guarantee that the invested time and money to bring a concept to maturity will be enough to solicit the funding to bring a design to fruition. Much time and energy has been expended to formalize the mission formulation phase and improve the process so that cost and complexity growth do not subject selected missions to delays or spending cuts, with Jet Propulsion Laboratory (JPL) being an industry leader in development of formulation tools and resources for early life cycle activities (Leising et al., 2010). Clearly, the cost serves as the primary constraint within which these exercises are facilitated, and the following section will outline this relationship and how it is approached in industry further.

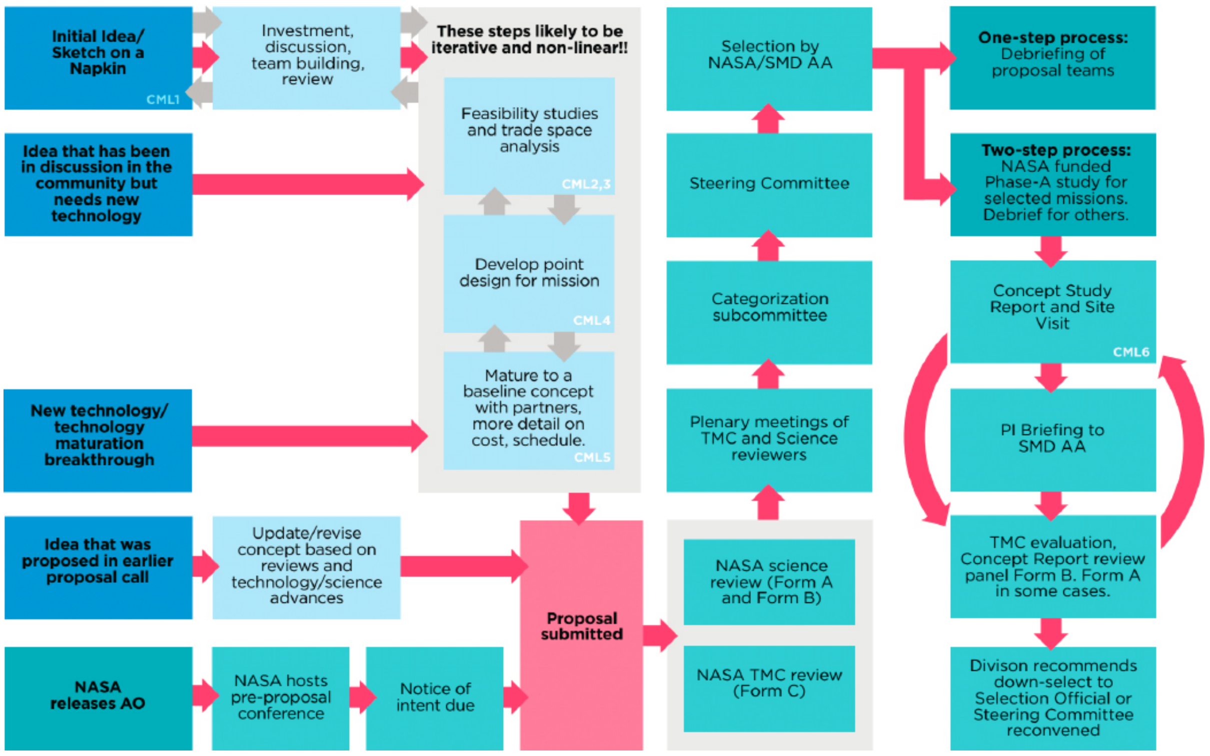

Initial Idea/ Sketch on a Napkin
CML1
Investment, discussion, team building, review
These steps likely to be iterative and non-linear!!
Feasibility studies and trade space analysis
CML2,3
Develop point design for mission
CML4
Mature to a baseline concept with partners, more detail on cost, schedule.
CML5
Idea that has been in discussion in the community but needs new technology
New technology/ technology maturation breakthrough
Idea that was proposed in earlier proposal call
Update/revise concept based on reviews and technology/science advances
NASA releases AO
NASA hosts pre-proposal conference
Notice of intent due
Proposal submitted
Selection by NASA/SMD AA
Steering Committee
Categorization subcommittee
Plenary meetings of TMC and Science reviewers
NASA science review (Form A and Form B)
NASA TMC review (Form C)
One-step process: Debriefing of proposal teams
Two-step process: NASA funded Phase-A study for selected missions. Debrief for others.
Concept Study Report and Site Visit
CML6
PI Briefing to SMD AA
TMC evaluation, Concept Report review panel Form B. Form A in some cases.
Divison recommends down-select to Selection Official or Steering Committee reconvened

Figure 3.1: Proposal Generation Process in Formulation (National Academies of Sciences, Engineering, and Medicine, 2022)

3.2 Cost Estimation

Considering the American context, NASA has seen a steady decline in budget as a proportion of government spending since the end of the Apollo era. Furthermore, the interplay of politics, election cycles, and public opinion make it such that NASA is often subject to volatile spending directives, impacting the ability to develop new projects and sometimes halting the progress of projects that have already been selected (Conley and Cobb, 2012). These factors may appear far removed from mission formulation, however they certainly demonstrate the motivation from an organizational perspective to most effectively utilize resources to develop economically feasible plans.

A topical example of this is the recently scrutinized Mars Sample Return (MSR) mission. This ambitious endeavour requires sending a spacecraft to Mars with necessary robotics to retrieve samples collected by the Perseverance rover, followed by a return phase in which those samples are brought safely to Earth for scientific analysis. Recently, an Independent Review Board (IRB) found that the program was projected to be severely over budget and requires significant design changes to mitigate the growing cost (JPL, 2023). Consequentially, the federal budget has allocated a smaller amount of funds to NASA than the agency requested, threatening both jobs and critical activities of other unrelated projects in development or seeking to being development. While this example is extreme and not the best indicator of the scope and cost of typical aerospace missions, it serves as a jarring reminder of the consequences that arise when cost grows well beyond that which is predicted in the early life cycle.

Furthermore, the status of MSR brings to light another inherent challenge cost engineers face, as they are often tasked with developing estimates for missions and objectives that have never been accomplished before. Sitting on the cutting edge of technology development and engineering progress, proposal generating teams may find themselves lacking any prior data points to effectively utilize a parametric scheme. In these cases, estimation by analogue is a common practice, by piecing together aspects of one or multiple missions that share attributes such as target, instrumentation, or power (as a short list of examples). Additional effort has been put into building non-parametric tools, such as the NASA Analogy Software Cost Model (ASCoT), which leverages KNN and spectral clustering to group missions in similarity based on limited and simple customer inputs reflective of the uncertainty of design specifics (Hihn et al., 2016).

Parametric analysis also serves a role in these exercises, however is often simple in nature such as linear regression. Engineers aim to learn cost estimating relationships (CERs) that can be used to create estimates provided inputs like those mentioned above, being either technical or design oriented in nature (Net et al., 2014).

In all cases, cost engineers must work diligently to extract as much information as possible out of the limited data in hand, presenting an opportunity to leverage generative methods for data augmentation.

CHAPTER 4

Experiments and Analysis

4.1 About the Data

The dataset of interest for this thesis was sourced from the Planetary Exploration Budget Dataset (The Planetary Society, 2023), which hosts a relational data sheet that has the yearly spending for NASA planetary science missions and activities. The data of interest came from one sheet in particular, "Timeline", which houses records at the mission level with features that correspond to mission costs, characteristics, and objectives. Additionally, a handful of columns pertaining to technical parameters were added to the sheet, namely the spacecraft mass, power, and number of instruments. All of these values were sourced from NASA or NASA affiliated sources, such as the National Space Science Coordinated Archive (NASA, 2022), in order to maintain consistency with the original sourcing. Furthermore, some columns were removed in an effort to simplify the dataset as much as possible to best replicate the broad trade space in which cost estimation is performed. The final pre-processing step consisted of taking the log-transformation of the mass and power variables, in order to reduce skewness present in the original distribution and to make the analysis less sensitive to outliers, as the raw values had wide domains with highly asymmetrical histograms. This dataset consists of 75 observations, for which a more complete description is provided for in table 4.1. The columns corresponding to formulation start and the launch are not considered important predictors for downstream analyses, but were included due to the formulation start containing missing values, and the sequential relationship between these variables in time.

Variable	Description	Type
Formulation Start	Fiscal year in which development began	integer
Launch Date	Fiscal year in which launch occurred	integer
Program Line	Program office which provided funding	string
Launch Vehicle	Rocket which carries scientific payload	string
LV Cost (2022)	Launch Vehicle cost inflated to FY22	float
Total Cost (2022)	Total cost inflated to FY22	float
Log Mass	Log-transformed spacecraft mass (in kg)	float
Log Power	Log-transformed spacecraft power (in W)	float
Number of Instruments	Count instruments on scientific payload	integer
Managing Center	Research center managing project	string
Destination Category	Classification of solar system target	string

Table 4.1: Descriptive summary of post-processed Planetary Budget dataset

4.2 Experiment Setup

Here we will discuss the experiment setup and goals for the analysis to follow on the modified Planetary Science Budget dataset. First, we consider generating data using DataSynthesizer as well as CuratedLLM, comparing the similarity of the data and the nature of any conditional relationships. To better assess the utility, we employ multiple machine learning tasks that correspond to the types of inference that are standard in cost estimation procedure for aerospace missions. This will include a linear regression task focused specifically on estimating cost the main variable of interest, as well as a classification task seeking to predict the managing center. When using the DataSynthesizer module, we consider 4 models: independent attribute, correlated attribute with 1 parent, correlated attribute with 2 parents, and correlated attribute with 3 parents. Similar, with CuratedLLM we consider both curated and uncurated data for comparison. The data was Curated with the same mechanism recommended in the paper, by training 100 iterations of an XGBoost classifier with default parametrization on the real data. Finally, CTGAN is included merely as a demonstration that traditionally deeper models are feasible generators in such as low-data setting. In all cases, 75 samples were generated, the same number of samples we have in the original data. The curation process led to discarding 16 samples that displayed low confidence and low uncertainty simultaneously ($\tau_{conf} < 0.2$ and $\tau_{aleatoric} < 0.15$).

4.3 Results and Comparison

4.3.1 Fidelity

As an initial model selection and fidelity test, we conduct quality analysis as set forth in the Python `SDV` (Synthetic Data Vault) package (Patki et al., 2016) as well as `DataSynthesizer` (Ping et al., 2017). The `DataSynthesizer` contains useful visualizations of correlations between attributes, as well as describing the Bayesian network computed when in correlated attribute mode. `SDV` serves as a more general provider for pre-processing, generation, and evaluation on multiple forms of data with many state of the art methods included, such as CTGAN. The `SDV` quality results are contained in Table 4.2 for data generated with all the considered models, with bold numbers denoting the best results for each respective metric.

Model/Metric	Column Shapes	Column Pair Trends	Overall Score
DataSynthesizer IAM	**89.58**	67.82	78.7
DataSynthesizer CAM $k = 1$	89.46	80.98	85.22
DataSynthesizer CAM $k = 2$	89.19	86.9	**88.05**
DataSynthesizer CAM $k = 3$	89.06	**86.99**	88.02
ChatGPT 4.0 Uncurated	79.36	78.14	78.75
ChatGPT 4.0 Curated	77.2	73.52	75.36
CTGAN	67.48	60.6	64.04

Table 4.2: SDV Quality Metric Results

When considering the differences in initial performance based on the above table, the generation philosophy is exceptionally apparent. One of the key aspects in the CuratedLLM paper is the demonstration that LLM-generated samples can extrapolate to unseen regions of the manifold. As such, it follows that samples generated by ChatGPT may express flexibility and deviation from observed distributions relative to DataSynthesizer, which is guaranteed to be within the observed bounds in sampling. This effect can potentially be observed, as the curation mechanism used appeared to negatively impact the quality evaluation from the perspective of the `SDV` package. However, we cannot rigorously test this as the entire dataset is used to generate our data, and we could not afford to hold out a large portion of observations as the CuratedLLM experimental setup does. A further note on the different DataSynthesizer methods is the fact that increasing the Bayesian Network complexity comes with diminishing returns, as the data simply is not large enough to learn deeper DAGs.

The observed histograms relative to the original data are shown in Figures 4.1 and 4.2 for the correlated attribute data with 2 parents, and the curated ChatGPT data.

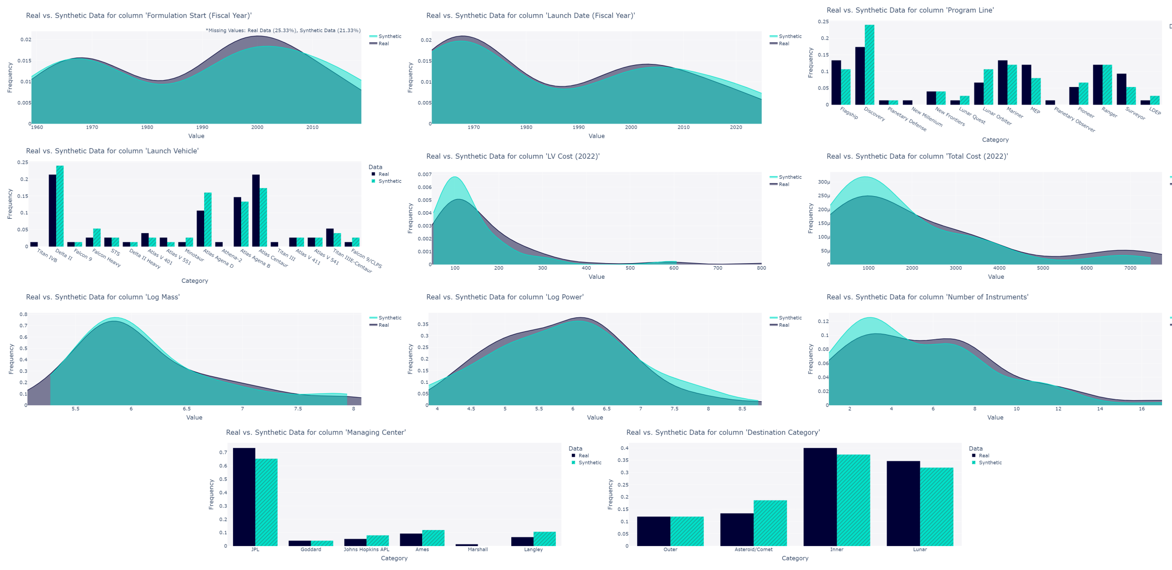


Figure 4.1: DataSynthesizer, correlated attribute mode, $k = 2$ Marginal Densities/Histograms

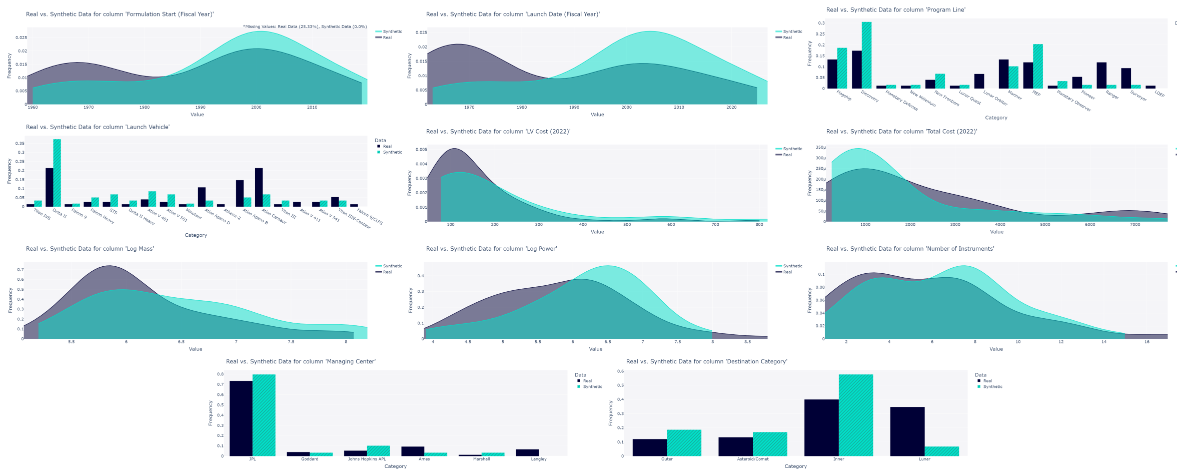


Figure 4.2: Curated ChatGPT Generation Marginal Densities/Histograms

The histograms are informative, particularly with regards to missing information. DataSynthesizer works by calculating a proportion of missing values in each attribute

(Formulation Start in this case), and reflects this in the sampling. Data generated by ChatGPT however did not contain any missing values, and the black box nature of the generation makes understanding how or if the missing data impacted generation decisions intractable. The missing values in the Formulation column did primarily correspond to the earliest program lines and missions, for which the general quality of data and depth of records do not necessarily exist. Therefore, it appears reasonable to speculate that ChatGPT implicitly generated samples with more recent dates perhaps in response to the absence of complete entries for the oldest missions.

Additionally, the form of the Bayesian networks used in the DataSynthesizer module is worth analyzing. The GreedyBayes algorithm does not account for implicit constraints that exist in the original data generating mechanism and enforce directional constraints on the generated DAG. As an example of this, consider the relationship between technical parameters and cost. From the perspective of a systems engineer, it is the technical parameters that drive cost, indicating a directionality from something like mass to cost. Understanding that cost is often the final target of analysis, it would be counter-intuitive to build data from a network that treats cost as a parent of the technical parameters. Therefore, simply generating a deeper more complex network does not inherent make the generated data better from a fidelity perspective. The generating mechanism should be justifiable to cost and systems engineers and as such reflect truthful, intuitive conditional relationships. Unfortunately, the black-box nature of LLM generation makes it such that conditional relationships cannot be directly inferred or visualized as a DAG, but these relationships can otherwise be inferred from pairwise attribute plots such as in Figures 4.3 and 4.4, which compare the impacts of curation on data quality.

Lastly, Figure 4.5 presents a visual representation of one of the DAGs created by DataSynthesizer. The treatment of mass as the root node through which the network is built is a particularly encouraging result that aligns with the contextual intuition of technical parameters driving cost and design decisions made further down the chain of mission formulation.

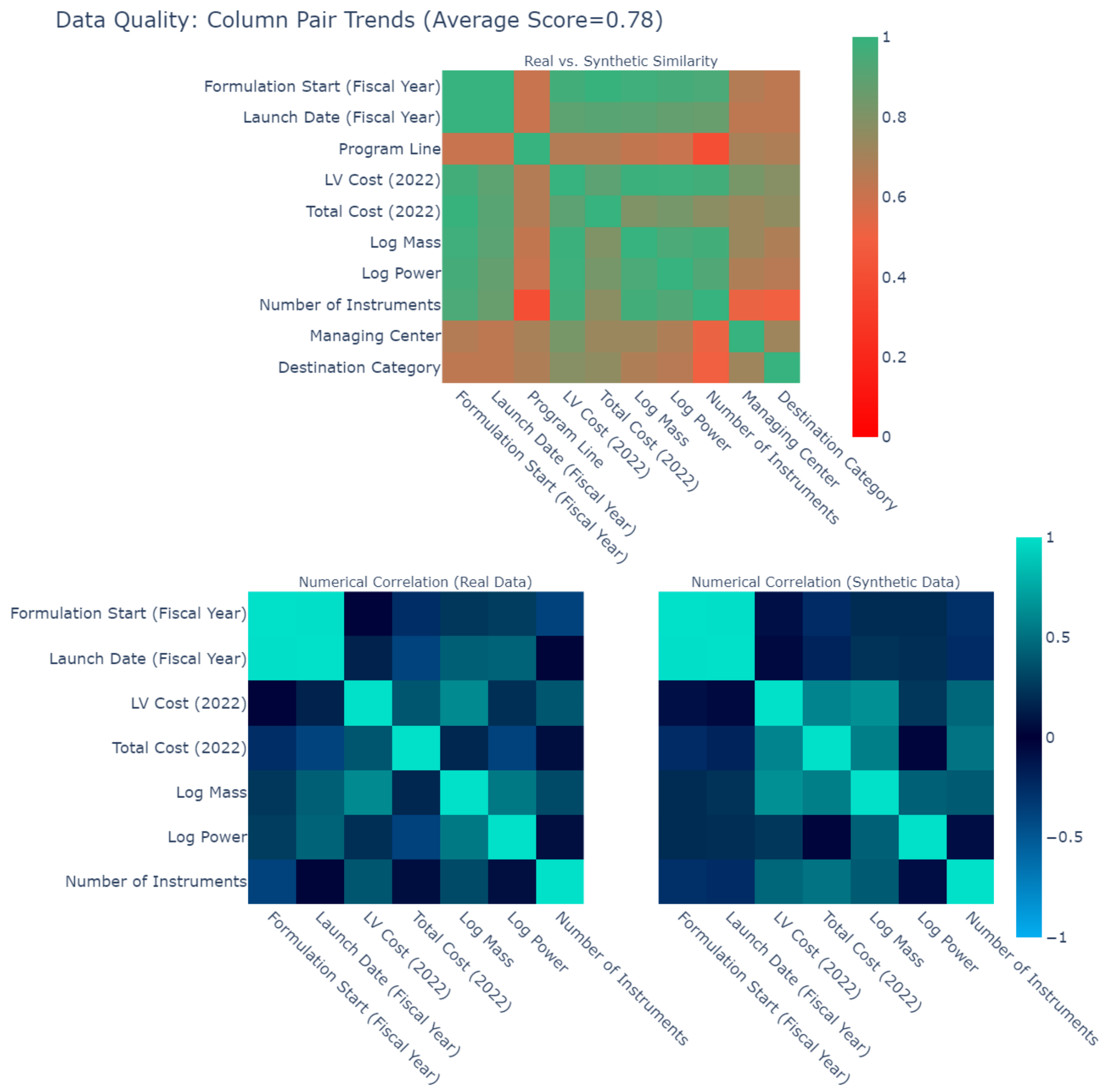


Figure 4.3: Real vs Synthetic Pairwise Correlations; Uncurated GPT data

4.3.2 Utility

We focus on a subset of simple machine learning tasks in order to assess the utility of the data generated by different models. The first task of interest is conducting a linear regression experiment with the goal of predicting cost.

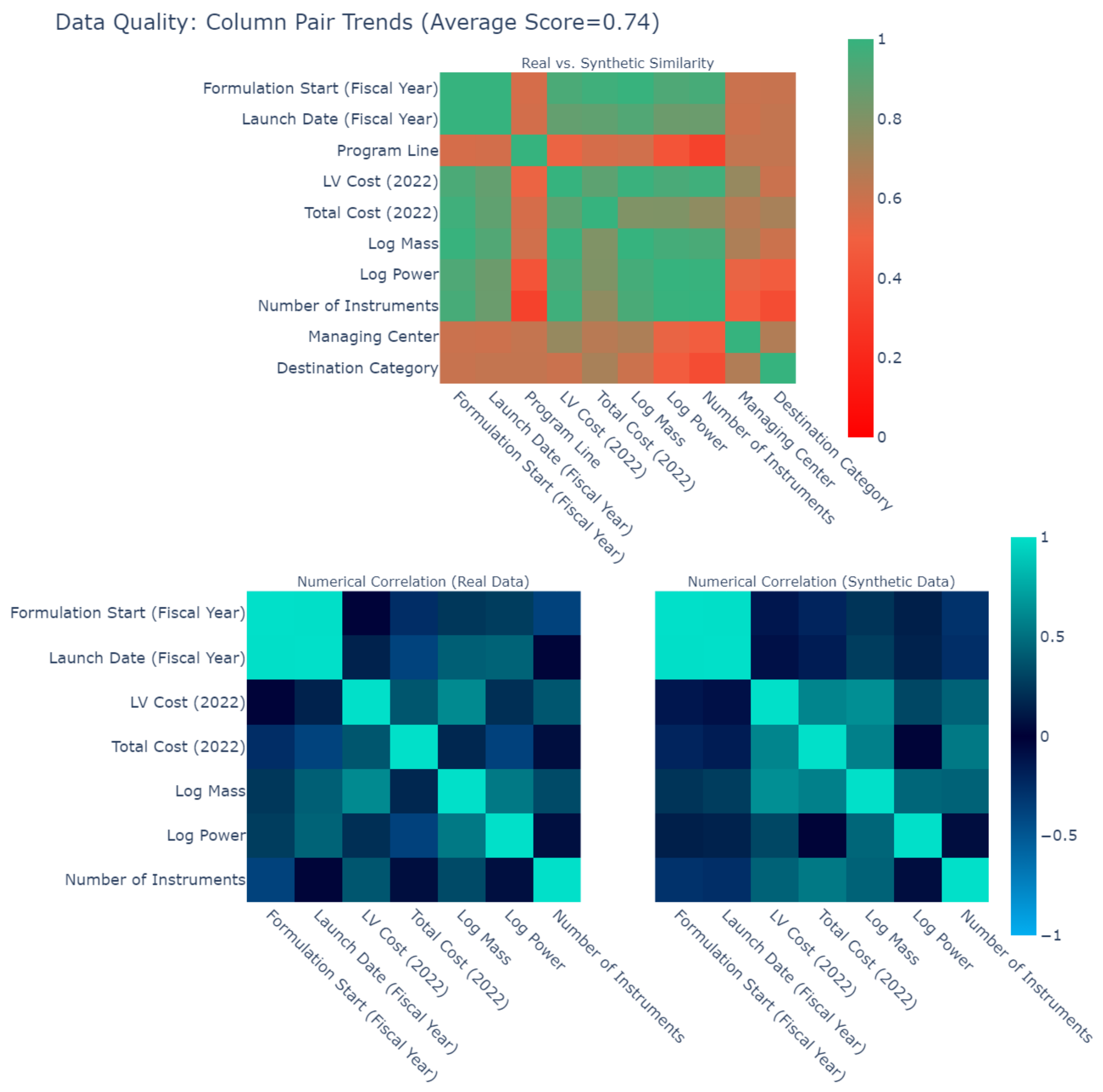


Figure 4.4: Real vs Synthetic Pairwise Correlations; Curated GPT data

The regression equation in question is the following:

$$\text{Total Cost (2022)} = \beta_0 + \beta_1 \log(\text{Mass}) + \beta_2 \text{Destination Category}$$

Table 4.3 contains many relevant metrics with regards to the regression results, including the R^2 metric, training error, test error, and parameter estimates. These analyses were conducted considering the Train on Synthetic Test on Real (TSTR) paradigm

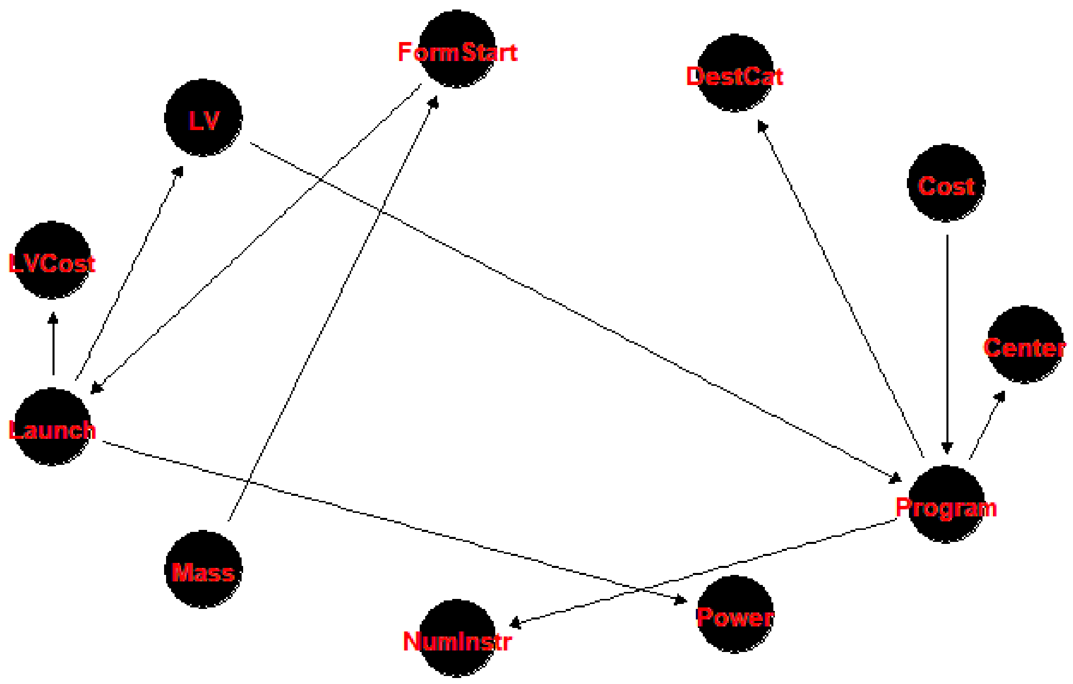


Figure 4.5: DAG Generated by Correlated Attribute mode, $k = 1$

(Esteban et al., 2017). This is used because ultimately it is the real data that we seek to make conclusions about, which we must not lose sight of. The synthetic data is simply a tool that permits more freedom with the real data, as the entire original dataset can be considered for model testing and validation.

Training Data	Testing Data	R^2	Train Error	Test Error
Real Data (full)		0.530	2,083,855	
Real Data (67%)	Real Data (33%)	0.575	1,737,614	2,887,323
DataSynthesizer IAM	Real Data	0.059	5,471,120	3,831,518
DataSynthesizer CAM $k = 1$	Real Data	0.373	2,937,660	2,582,095
DataSynthesizer CAM $k = 2$	Real Data	0.466	1,618,692	2,304,025
DataSynthesizer CAM $k = 3$	Real Data	0.469	1,617,887	2,303,361
ChatGPT 4.0 Uncurated	Real Data	0.504	1,311,656	3,186,160
ChatGPT 4.0 Curated	Real Data	**0.663**	**1,021,134**	**2,137,181**

Table 4.3: Regression Results

We utilize a KNN based methodology to conduct our classification task, attempting to classify the Managing Center as either "JPL" or "Other" - ie a binary classification problem. The "Managing Center" column does by default have > 2 classes, however

JPL is the most dominant class and all others were coerced to "Other" in an effort to simplify the curation mechanism. This classification task has an identical variable of interest to that which was used to conduct the data curation step for the data generated by ChatGPT, though we use KNN instead of XGBoost due to the prevalence in existing industry methodology. 5 neighbors are used for all models. Table 4.4 contains pertinent information regarding model benchmarking under this setting.

Training Data	Testing Data	Train Accuracy	Test Accuracy
Real Data (full)		0.88	
Real Data (67%)	Real Data (33%)	0.865	0.789
DataSynthesizer IAM	Real Data	0.893	0.92
DataSynthesizer CAM, $k = 1$	Real Data	0.88	0.786
DataSynthesizer CAM, $k = 2$	Real Data	0.96	0.907
DataSynthesizer CAM, $k = 3$	Real Data	0.96	**0.947**
ChatGPT 4.0 Uncurated	Real Data	0.947	0.907
ChatGPT 4.0 Curated Binary	Real Data	**1.0**	**0.947**

Table 4.4: KNN Results

In both machine learning settings, the curated dataset generated by ChatGPT exceeds all other methods with regards to standard error and accuracy metrics (MSE and 0-1 Loss). That being said, the curated method provides a higher R^2 value than a model trained on the real data, and caution must be exercised such that noise in the generative process does not get mistaken for true signal, as it could lead to reinforcement of relationships that are not true to the real world data. These findings are true despite the fact that the curated dataset is smaller than all other generated datasets, lending credibility to the curation process removing noisy and low-quality samples. Furthermore, these findings indicate that dataset curation can have positive downstream implications for models outside of the classification task used in curation, though further analysis is required before drawing general conclusions about the applicability of this method.

Overall, the synthetic data generated appears to demonstrate reasonable fidelity and utility, particularly when considering correlated attributes when using DataSynthesizer, and the CuratedLLM. The curated dataset led to a regression model with lower training and testing error than analysis on just the real data alone, and similarly achieved perfect train accuracy and near perfect test accuracy on a clustering classification task. Both of these facts prove the effectiveness of the synthetic data, as it permits the creation of models that are similar to models trained on real data, while freeing the real data for testing and validation use.

CHAPTER 5

Conclusion

In this thesis, we consider the ability of generative data science models to produce trustworthy samples in the context of aerospace industry developmental activities. We approached the problem from the generative perspective, we surveyed the current state of the literature and existing mainstream methods, while focusing on methods and philosophies that support the low-data paradigm. Similarly, we built the motivation from the perspective of a cost or systems engineer in an aerospace organization, understanding the processes that immature mission concepts go through and how cost estimation serves as a scenario exploration exercise.

Further, we specifically address the fidelity and utility of generated data via both visual and analytic means, comparing the features of the synthetic data to the original as well as the possibility for increased model performance on simple downstream machine learning tasks.

This line of research presents a number of opportunities to continue work and further explore techniques to optimize generative data performance. Simple experimental extensions include varying the amount of data generated, for example generating datasets orders of magnitude larger than the original, while comparing the balance between generalizability and overfitting, as well as applying the curation mechanism to all datasets. The choice of generating 75 data points was an arbitrary one chosen to mimic the real data in a one-to-one manner. However, with a reliable generator, there is no reason that hundreds, if not thousands of synthetic samples can be created in the same way. That being said, particularly when considering parametric analysis there is important information not only carried by a parameter estimate, but also by the uncertainty. Generating unreasonable numbers of points and increasing the size of the training data will serve to lower parameter uncertainty, which may not be a desired goal depending on the task. Further research into the field of generative data science and impacts on uncertainty quantification is needed. Exploring further downstream model structures and how they are impacted by generation methods could help lend further credibility to the potential synthetic data has to augment cost estimation activities. Another natural extension of this work would be to incorporate differential privacy constraints to facilitate a data sharing scenario, and analytically identifying the impact the trade-off between utility and privacy when in such an extreme low-data setting. As stated prior, proposal generation is a competitive process, and as such data sharing between aerospace institutions is subject to significant regulations, presenting a different but tangentially related use case that merits some investigation.

Ultimately, the biggest takeaway we hope to present is that industry leaders can and should be excited by the advances in data generation and management and that it is relevant to all forms and sizes of data. The academic sector and industry sector should have a symbiotic relationship, in which use cases such as that which we have outlined in this thesis motivate application and evaluation of novel technologies to bridge the gap between theory and practice.

REFERENCES

Borisov, V., Seßler, K., Leemann, T., Pawelczyk, M., and Kasneci, G. (2023). Language Models are Realistic Tabular Data Generators. 15

Chen, T. and Guestrin, C. (2016). XGBoost: A Scalable Tree Boosting System. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM. 16

Conley, R. S. and Cobb, W. W. (2012). Presidential Vision or Congressional Derision? Explaining Budgeting Outcomes for NASA, 1958–2008. 20

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. 14

Dwork, C. (2008). Differential Privacy: A Survey of Results. In Agrawal, M., Du, D., Duan, Z., and Li, A., editors, *Theory and Applications of Models of Computation*, pages 1–19, Berlin, Heidelberg. Springer Berlin Heidelberg. 9

Dwork, C. and Roth, A. (2014). The Algorithmic Foundations of Differential Privacy. 9

Esteban, C., Hyland, S. L., and Rätsch, G. (2017). Real-valued (Medical) Time Series Generation with Recurrent Conditional GANs. *arXiv e-prints*, page arXiv:1706.02633. 29

Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative Adversarial Networks. 6

Heilweil, R. and Alder, M. (2023). Generative Ai isn't quite cleared for takeoff at NASA. `https://fedscoop.com/generative-ai-isnt-quite-cleared-for-takeoff-at-nasa/`. 17

Hihn, J., Juster, L., Johnson, J., Menzies, T., and Michael, G. (2016). Improving and expanding NASA software cost estimation methods. In *2016 IEEE Aerospace Conference*, pages 1–12. 20

Jordon, J., Yoon, J., and van der Schaar, M. (2018). PATE-GAN: Generating Synthetic Data with Differential Privacy Guarantees. In *International Conference on Learning Representations*. 7

JPL (2023). NASA releases Independent Review's Mars sample return report. `https://www.jpl.nasa.gov/news/nasa-releases-independent-reviews-mars-sample-return-report`. 20

Kotelnikov, A., Baranchuk, D., Rubachev, I., and Babenko, A. (2023). TabDDPM: Modelling Tabular Data with Diffusion Models. In Krause, A., Brunskill, E., Cho, K., Engelhardt, B., Sabato, S., and Scarlett, J., editors, *Proceedings of the 40th*

International Conference on Machine Learning, volume 202 of *Proceedings of Machine Learning Research*, pages 17564–17579. PMLR. 8

Leising, C. J., Sherwood, B., Adler, M., Wessen, R. R., and Naderi, F. M. (2010). Recent improvements in JPL's mission formulation process. In *2010 IEEE Aerospace Conference*, pages 1–12. 19

McKenna, R., Miklau, G., and Sheldon, D. (2021). Winning the NIST Contest: A scalable and general approach to differentially private synthetic data. 5

NASA (2022). Welcome to the NSSDCA. `https://nssdc.gsfc.nasa.gov/`. 22

National Academies of Sciences, Engineering, and Medicine (2022). *Advancing Diversity, Equity, Inclusion, and Accessibility in the Leadership of Competed Space Missions*. The National Academies Press, Washington, DC. vi, 18, 19

Net, M. S., Selva, D., and Golkar, A. (2014). Exploring classification algorithms for early mission formulation cost estimation. In *2014 IEEE Aerospace Conference*, pages 1–14. 21

OpenAI (2024). GPT-4 Technical Report. 1

Patki, N., Wedge, R., and Veeramachaneni, K. (2016). The Synthetic data vault. In *IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, pages 399–410. 24

Ping, H., Stoyanovich, J., and Howe, B. (2017). DataSynthesizer: Privacy-Preserving Synthetic Datasets. In *Proceedings of the 29th International Conference on Scientific and Statistical Database Management*, SSDBM '17, New York, NY, USA. Association for Computing Machinery. 6, 11, 24

Rubin, D. B. (1993). Discussion : Statistical Disclosure Limitation. 5

Seedat, N., Huynh, N., van Breugel, B., and van der Schaar, M. (2024). Curated LLM: Synergy of LLMs and Data Curation for tabular augmentation in ultra low-data regimes. 13

The Planetary Society (2023). The Planetary Exploration Budget Dataset. `https://www.planetary.org/space-policy/planetary-exploration-budget-dataset`. 22

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2023). Attention Is All You Need. 14

Wessen, R. R., Borden, C., Ziemer, J., and Kwok, J. (2013). Space mission concept development using concept maturity levels. 18

Xie, L., Lin, K., Wang, S., Wang, F., and Zhou, J. (2018). Differentially Private Generative Adversarial Network. 7

Xu, L., Skoularidou, M., Cuesta-Infante, A., and Veeramachaneni, K. (2019). Modeling Tabular data using Conditional GAN. 7

Zhang, J., Cormode, G., Procopiuc, C. M., Srivastava, D., and Xiao, X. (2017). PrivBayes: Private Data Release via Bayesian Networks. *ACM Trans. Database Syst.*, 42(4). 4, 6

Zhang, Z., Li, M., and Yu, J. (2018). On the convergence and mode collapse of GAN. In *SIGGRAPH Asia 2018 Technical Briefs*, SA '18, New York, NY, USA. Association for Computing Machinery. 7