Bachelor's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Computer Science

Aleksander Brekke Røed
Erlend Rønning

Azure Cloud Cost Prediction: Monolithic and Multi-Model Approaches

Bachelor's thesis in Computer Science
Supervisor: Ole Christian Eidheim
May 2023

NTNU
Norwegian University of
Science and Technology

Aleksander Brekke Røed
Erlend Rønning

Azure Cloud Cost Prediction: Monolithic and Multi-Model Approaches

Bachelor's thesis in Computer Science
Supervisor: Ole Christian Eidheim
May 2023

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Computer Science

NTNU
Norwegian University of
Science and Technology

Abstract

In this bachelor's thesis, we explore the use of a monolithic and a multi-model machine learning approach to predict cloud pricing for Azure services, focusing on custom-tailored models for individual customers. The research explores various factors influencing cloud prices such as type, region, demand, and market trends. The goal is to explore and determine the effectiveness of these machine learning models in the context of SkyeTec AS's customer data. By developing accurate and reliable cloud pricing prediction models, the study aims to contribute insights into the area of cloud cost prediction, potentially helping in the optimization of cloud expenses.

Sammendrag

I denne bacheloroppgaven utforsker vi bruken av en monolittisk og en multimodell maskinlæringstilnærming for å forutsi priser til Azure-tjenester, med fokus på skreddersydde modeller opp mot enkeltkunder. Forskningen undersøker ulike faktorer som påvirker skypriser, for eksempel type, region, etterspørsel og markedstrender. Målet er å utforske og bestemme effektiviteten til disse maskinlæringsmodellene i sammenheng med SkyeTec AS' kundedata. Ved å utvikle nøyaktige og pålitelige modeller for prediksjon av skypriser tar studien sikte på å bidra med innsikt i området for prediksjon av skykostnader, og potensielt bidra til optimalisering av skyutgifter.

Preface

This bachelor thesis was written on behalf of SkyeTec AS by two computer science students at The Norwegian University of Science and Technology. The thesis explores the viability of two approaches to cloud cost prediction using machine learning, and builds on knowledge accumulated through several relevant courses and side projects.

The task was chosen due to the exciting prospect of working with machine learning, which both members had experience with. The team also wanted to work with a competent client, and this task met both requirements. However, none of the team members had any experience working with cloud computing, something that made the initial research long and intensive to make sure the team had a comprehensive understanding of the task before attempting to solve it.

Through the process of writing this bachelor's thesis, the team has reaped the rewards of structuring its approach to the task, a well-flowing team dynamic, and guidance from both client and supervisor.

The Git repository of the thesis is private due to competitive considerations and the dataset is not appended due to privacy concerns of the customer. The source code can be found in attachment 2.

We would like to thank our supervisor Ole Christian Eidheim for constructive feedback as well as for pushing the team to make this thesis something we are proud of. We would also like to thank Heidi Kristin Bjørnslett Haugen, CEO of SkyeTec AS, for allowing us to write this thesis on behalf of the company. Lastly, we would like to thank Magne Worren, our technical contact at SkyeTec AS, for his invaluable insight on cloud computing, the dataset, and general feedback on the thesis.

Aleksander Brekke Røed

Erlend Rønning

Contents

List of Figures

List of Tables

Abbreviation	Full form
RF	Random Forest
KNN	K-Nearest Neighbors
MLP	Multi-Layer Perceptron
Ridge	Ridge Regression
GBR	Gradient Boosting

Table 1: Abbreviations of model types used in figures and tables.

Shortened form	Full form
Storage	microsoft.storage/storageaccounts
Disks	microsoft.compute/disks
VMs	microsoft.compute/virtualmachines
Recovery vaults	microsoft.recoveryservices/vaults
VM scls	microsoft.compute/virtualmachinescalesets
Key vaults	microsoft.keyvault/vaults
Load balancers	microsoft.network/loadbalancers
Public IP	microsoft.network/publicipaddresses
Workspaces	microsoft.operationalinsights/workspaces
Bastion	microsoft.network/bastionhosts
Private DNS	microsoft.network/privatednszones
VM network	microsoft.network/virtualnetworkgateways
Network watchers	microsoft.network/networkwatchers
Galleries	microsoft.compute/galleries
Namespaces	microsoft.servicebus/namespaces
Pricings	microsoft.security/pricings
Containers	microsoft.containerinstance/containergroups
Firewalls	microsoft.network/azurefirewalls
Connections	microsoft.network/connections
Registries	microsoft.containerregistry/registries
Action groups	microsoft.insights/actiongroups
Express routes	microsoft.network/expressroutecircuits
Metric alerts	microsoft.insights/metricalerts
Snapshots	microsoft.compute/snapshots
Images	microsoft.compute/images
Restore points	microsoft.compute/restorepointcollections
Private endpoints	microsoft.network/privateendpoints
Scheduled query rules	microsoft.insights/scheduledqueryrules

Table 2: Shortened form of Microsoft Azure resource types used in figures and tables.

Acronyms

ADWIN ADaptive WINdowing. vi, 10, 19, 30, 33

AI Artificial Intelligence. 1, 5, 12

API Application Programming Interface. 13, 33

ARIMA Autoregressive Integrated Moving Average. 33

AWS Amazon Web Services. iv, 1, 3, 10, 31–34

DSR Design Science Research. vi, 16, 17, 30, 31

GB Gradient Boosting. vi, 8, 21

GCP Google Cloud Platform. 1, 3

IaaS Infrastructure-as-a-Service. 3, 4, 18

IDE Integrated Development Environment. 12

KNN k-Nearest Neighbors. 9, 22

MAE Mean Absolute Error. 10

ML Machine Learning. iv, 1, 5, 6, 17, 33

MLP Multi-Layer Perceptron. vi, 9, 21

MSE Mean Squared Error. 7, 8, 10, 11, 20, 22, 24, 26, 33

NN Neural Network. 9

PaaS Platform-as-a-Service. 3, 4, 18

PAYG Pay-as-you-go. 4, 5, 18

RF Random Forest. vi, 8, 20, 21

RIs Reserved instances. 4, 18

R^2 R-squared. 7, 20, 22, 24, 26, 30, 32, 33, 35

SaaS Software-as-a-Service. 3, 4

SARIMA Seasonal Autoregressive Integrated Moving Average. 11, 15, 33, 34

sklearn Scikit-learn. 13, 19, 22

SLR Systematic Literature Review. v, 10, 15, 16, 30

VMs Virtual Machines. 3, 4, 32, 33

1 Introduction

1.1 Background and Motivation

Cloud computing has revolutionized the way businesses manage their digital infrastructure, providing flexible, scalable resources and cost-effective solutions. Major companies in the cloud services market including Amazon Web Services (AWS), Microsoft Azure, and Google Cloud Platform (GCP), provide several services for computational needs [1].

However, with the growing complexity and usage of cloud services, understanding and managing costs has become a challenging task. This has increased the need for effective tools and methods to predict and control cloud expenses. Machine Learning (ML), a subfield of Artificial Intelligence (AI), has shown promising results in this regard [2]. An advantage of ML algorithms is their ability to understand intricate patterns in data and make accurate predictions based on them [3].

SkyeTec AS, a company with strong competence in cloud services, recognized the need for effective cloud cost management. Since March 2020, the company has been involved in several projects involving the design, architecture, and implementation of cloud solutions, primarily based on Microsoft Azure and Amazon Web Services (AWS) [4].

Understanding the need for effective cost management tools, SkyeTec AS has plans to build a portal for its customers. This portal aims to provide predicted values of future cloud expenses. However, the field of custom-tailored cloud cost prediction is relatively unexplored and poses challenges.

1.2 Aim of the Study

The primary aim of this bachelor's thesis is to explore the use of machine learning to predict cloud costs for Azure services. Using data from a SkyeTec AS customer, the study focuses on developing two machine-learning approaches: a monolithic approach and a multi-model approach. The monolithic approach uses one model and treats the entire cost data as a single entity, while the multi-model approach uses individual models for each distinctive resource type. This research aims to analyze the performance of these models, providing insights into their effectiveness in predicting cloud costs.

1.3 Structure

The structure of this bachelor's thesis is organized as follows:

Introduction - This chapter provides an overview of the research topic, including the background, motivation, and aim of the study.

Theory - This chapter contains the theoretical concepts relevant to the research and gives the foundation to understand the method and discussion part of the report.

Technology - This chapter provides an in-depth exploration of the technological tools and platforms used in the research.

Method - This chapter presents the research methodology and how the theory has been implemented using the technology.

Results - This chapter presents the results obtained from the data and machine learning models when following the methodology explained in the method chapter.

Discussion - This chapter presents a discussion of the results.

Conclusion - This chapter summarizes the research, drawing conclusions from the findings and discussions in the other chapters. It also suggests potential areas for future research and improvements to the machine learning models.

2 Theory

This chapter contains the theoretical concepts relevant to the research and gives the foundation to understand the method and discussion part of the report.

2.1 Overview of cloud computing

Cloud computing is the delivery of on-demand computing resources over the internet (the cloud). This includes resources like storage, processing power, databases, and networking. Cloud solutions have been shown to be cost-effective, due to their high scalability, accessibility, and security [5] [6]. Compared to traditional on-premises digital infrastructure, cloud services offer a more versatile and modern approach to managing computing resources. Applications quickly can be deployed without big upfront investments in hardware and software. The biggest cloud service providers are Amazon Web Services (AWS) (33% market share), Microsoft Azure (22% market share), and Google Cloud Platform (GCP) (10% market share) [1].

2.1.1 Types of cloud service models

The three main types of cloud service models, or as-a-Service solutions, are Platform-as-a-Service (PaaS), Software-as-a-Service (SaaS), and Infrastructure-as-a-Service (IaaS), the key difference between them being the level of abstraction provided.

- **PaaS:** PaaS provides a hardware and software platform where users can run applications and store data the application relies on. This solution is primarily used by programmers and developers because it gives users a shared cloud development environment, which is an important part of DevOps.
- **SaaS:** SaaS is a service where a complete application is delivered to the user, typically accessed through a web browser. Zoom, Microsoft Office 365, and Google Workspace are examples of SaaS services. The provider manages security, bug fixes, and software updates, making these services convenient for organizations that want to focus on their core business activities.
- **IaaS:** With IaaS, organizations can provision Virtual Machines (VMs), storage, and other resources on demand. Users are also offered a wide range of monitoring and management tools for their virtual resources. These tools allow users to build and manage their own development and deployment environment.

IaaS offers virtualized infrastructure components including networking, storage, and VMs. PaaS offers a comprehensive platform for creating, testing, and deploying applications, replete with database management systems, runtime environments, and development tools. SaaS offers pre-configured software programs that can be accessed online [7].

2.1.2 Hierarchical structure of Microsoft Azure

Microsoft Azure employs a hierarchical structure to manage and organize cloud resources efficiently. At the highest level, subscriptions act as the primary billing and access control units,

under which multiple resource groups can be created. Resource groups are logical containers for resources that are deployed within an Azure subscription, allowing for organized management and monitoring. Within each resource group, resources such as VMs, storage accounts, and web apps are provisioned and managed, facilitating a streamlined approach to organizing and maintaining cloud infrastructure [8].

- **Resources:** Resources are the lowest abstraction level in the hierarchy of Azure and are individual instances of services or components in its ecosystem. Everything from storage and recovery vaults to firewalls and network watchers is managed as resources. For a complete list of the resource types covered in this study, see table 2.
- **Resource groups:** Resource groups are customizable logical containers for resources within a subscription. They allow related resources to be organized and managed together, as well as grant access control.
- **Subscriptions:** Subscriptions are the highest abstraction level mentioned in this study. They contain resource groups and typically power an app or some microservice.

2.1.3 Cloud pricing models

As the usage of cloud computing continues to grow, understanding the differences between cloud pricing models becomes important for users looking to maximize the value and efficiency of their cloud services.

- **Pay-as-you-go (PAYG):** The PAYG pricing model is recognized as one of the more flexible pricing options within cloud computing. Characterized by its consumption-based approach, this model ensures that users only pay for the resources that they utilize, contributing to its cost-effectiveness and adaptability [9].
- **Reserved instances (RIs):** RIs represents a potentially more cost-effective alternative. By committing to a predetermined plan, users are granted discounts on resources compared to the rates of the PAYG model [10].

These are the cloud pricing models used by SkyeTec AS, and are therefore the relevant ones. While other cloud pricing models are available on the market, a detailed exploration of these falls beyond the scope of the study.

2.1.4 Factors influencing cloud pricing

There are numerous factors deciding the final cost of cloud services. Understanding these factors and their impact is essential for creating a predictive model that accurately estimates cloud pricing. This section will discuss key factors that influence the pricing of cloud resources [11].

- **Service Model:** Cloud service providers offer various service models. These models, such as PaaS, SaaS and IaaS provides different pricing structure that influence the cost of cloud services.

- **Resource usage:** Resource usage is a significant aspect of cloud pricing. Cloud service providers frequently employ the PAYG model. Resource usage will therefore directly impact cost.
- **Data Storage and Transfer:** Cloud providers will typically charge customers based on the quantity of data stored, and transfer data rates. Different storage types, including object storage, block storage, and file storage may use different pricing structures.
- **Location and Availability Zones:** Cloud providers often operate numerous data centers, known as availability zones. These data centers can be located in various parts of the world. The cost can vary as electricity, real estate, and labor costs differ by region. Choosing to use multiple availability zones for redundancy and availability will also affect cost.
- **Scalability and Elasticity:** Scalability and elasticity enable users to alter their resource utilization based on demand. This can also have an impact on cloud pricing, as customers may choose to use auto-scaling features. This can increase or decrease resource usage, and as a result, cost.

2.2 Machine learning techniques

Machine Learning (ML) is a subfield of Artificial Intelligence (AI) that focuses on developing and optimizing algorithms to enable computers to learn from data. The goal of machine learning is for computers to make predictions and decisions based on similar data, without needing to explicitly program it to do so [3]. Because of the rise of Large Language Models and services like OpenAI's ChatGPT, ML and AI have become household terms in recent months [12].

2.2.1 Types of ML

Typically, ML is divided into three main types: supervised learning, unsupervised learning, and reinforcement learning. Each of these has its own approach to learning from data [3].

- **Supervised learning:** In supervised learning, the model is trained on a labeled dataset, which means each data point has an associated output value or class label. The objective is to learn a mapping from input features to output labels, allowing the model to make predictions for unseen data. Common supervised learning algorithms include linear regression, logistic regression, support vector machines, decision trees, and random forests. Supervised learning can be further divided into two categories:
    1. Regression: The goal is to predict a continuous output value based on input features. Examples include predicting housing prices, stock prices, temperature, or in this case cloud resource prices.
    2. Classification: The goal is to predict a discrete class label for a given set of input features. Examples include spam detection, image recognition, and diagnosing diseases.
- **Unsupervised learning:** Unsupervised learning involves training the model on a set of data points without associated output values or class labels. The objective is to find

hidden patterns, structures, or relationships within the data. Unsupervised learning algorithms can be used for clustering, dimensionality reduction, or generating new data samples. Common unsupervised learning algorithms include k-means clustering, hierarchical clustering, DBSCAN, principal component analysis, and t-distributed stochastic neighbor embedding.

- **Reinforcement learning:** In reinforcement learning, an agent learns to make decisions by interacting with an environment and receiving feedback through rewards or penalties. The goal is to learn how to maximize rewards over time. Reinforcement learning is especially suited for problems with no readily available optimal solution, and the agent must explore the environment and learn by doing. Common reinforcement learning algorithms are Q-learning, Deep Q-Networks, policy gradients, and Actor-Critic methods.

2.2.2 Key concepts

The following section will present relevant key concepts of ML.

- **Features and labels:** Features and labels play a crucial role in developing and evaluating models. Features, also known as input variables or attributes, represent the independent variables used by the model to make predictions. They can be numerical or categorical and are typically organized in a multi-dimensional feature space. Selecting relevant features and transforming or engineering new ones can significantly impact model performance.

  Labels, also known as target or output variables, represent the dependent variable the model aims to predict. In supervised learning, labels are provided as part of the training data, allowing the model to learn the relationship between input features and output labels. For classification tasks, labels are discrete class categories, while for regression tasks, they are continuous values.

  It is essential to carefully preprocess features and labels, ensuring that they are properly scaled, and encoded, in order to develop accurate and reliable machine learning models [13].

- **Overfitting and underfitting:** Overfitting is a common issue in machine learning where a model learns the training data too well, capturing noise or random fluctuations rather than the underlying patterns. As a result, the model becomes too complex and performs poorly on new, unseen data. This is because it has essentially memorized the training data rather than generalizing from it. Overfitting can be caused by several factors, including using a model that is too complex for the given data, having insufficient training data, or the presence of noisy or irrelevant features [14]. Conversely, underfitting occurs when a model adapts too slowly to the training data.

- **Training and testing:** In the context of developing a machine learning model, it is crucial to split the available data into separate training and test sets. The training set is used to train the model, allowing it to learn patterns and relationships between input features and target labels. The test set, on the other hand, is reserved for evaluating the model's performance on unseen data, providing an estimate of its generalization ability.

This separation helps assess how well the model can make predictions on new data and prevents overfitting.

- **Cross-validation:** Cross-validation is a data resampling method to assess the generalization ability of predictive models and to prevent overfitting [15]. This is a technique where the data is split into a predetermined number of folds. In each iteration, one fold is used as test data, while the remaining folds are used as training data. The model is evaluated on each fold, and using the average accuracy across all iterations provides a more reliable estimate of the generalization of the model.
- **Loss functions:** Loss functions, also known as cost functions or objective functions, are a fundamental component of machine learning algorithms, as they quantify the difference between the predicted outputs and the actual labels. By measuring the model's performance, loss functions provide a means to optimize the model during the training process. Different loss functions are suited for different tasks, such as Mean Squared Error (MSE) for regression or cross-entropy for classification. The choice of an appropriate loss function is crucial, as it determines how the model learns from the data and can significantly impact its performance. In many optimization algorithms, such as gradient descent, the goal is to minimize the loss function to achieve the best possible model for the given problem [16].
- **R-squared (R²):** The R² metric is a statistical measure of the goodness of fit of a regression model, indicating how well the variance in the dependent variable can be explained by the independent variables. In other words, R² quantifies how closely the training data fits the regression model, providing an assessment of the model's explanatory power.

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}i)^2}{\sum i = 1^n (y_i - \bar{y})^2} \tag{1}$$

In this formula, n represents the number of samples, y_i is the true target value for the i-th sample, $\hat{y}_i$ is the predicted target value for the i-th sample, and $\bar{y}$ is the mean of the true target values. The numerator represents the sum of squared residuals (errors), and the denominator represents the total sum of squares. The R² value is calculated as 1 minus the ratio of these two quantities [17].

2.2.3 Linear regression techniques

Linear regression is a fundamental machine learning technique used for predicting a continuous target variable based on one or more input features. It assumes a linear relationship between the input features and the target variable and estimates the model coefficients by minimizing the remaining sum of squares. Linear regression is easy to interpret, computationally efficient, and serves as a baseline model for more complex regression techniques. However, its simplicity can also limit its ability to model complex, non-linear relationships in data.

Ridge regression is one of the models used in this study. It uses regularization, that is adding a penalty term to the Ordinary Least Squares loss function to combat overfitting. The regularization technique used in ridge regression is called L2 [18].

$$L(\boldsymbol{w}) = \frac{1}{n}\sum_{i=1}^{n}(y_i - (\boldsymbol{w} \cdot \boldsymbol{x}i))^2 + \lambda \sum j = 1^p w_j^2 \tag{2}$$

Here, $L(\boldsymbol{w})$ represents the Ridge loss function, n is the number of training examples, y_i is the target value for the i-th example, $\boldsymbol{w}$ is the vector of model coefficients (weights), $\boldsymbol{x}_i$ is the vector of input features for the i-th example, p is the number of input features, and λ is the regularization parameter. The first term represents the mean squared error, while the second term is the L2 regularization term.

2.2.4 Ensebmle learning techniques

Ensemble learning techniques in machine learning are methods that combine multiple individual models, called base learners, to improve overall prediction performance. By aggregating the predictions of several base learners, ensemble learning techniques can reduce overfitting, increase model robustness, and achieve better generalization. Common ensemble methods include bagging, which builds multiple models independently and averages their predictions, and boosting, which builds models sequentially, with each new model focusing on correcting the errors of the previous ones. Ensemble techniques have been proven effective across a wide range of problems and often yield state-of-the-art results in both classification and regression tasks [19]. In this study, two ensemble techniques are used: Random Forest (RF) and Gradient Boosting (GB).

RF is an ensemble learning method that combines multiple decision trees to make predictions. A large number of individual decision trees are constructed during training, each built using an arbitrary subset of the training data. When predicting, the algorithm accumulates the outputs of all the decision trees and takes a majority vote for classification or averaging the predictions for regression tasks. Because of the way RF averages the results of its decision trees, the model reduces variance and potential biases associated with individual trees. This makes the model adept at capturing complex patterns while having a relatively low risk of overfitting [20].

GB is another ensemble learning method that combines multiple weak learners, usually decision trees. The model differs from RF in that it is sequential. Each of the learners attempts to correct the errors made by the one before it. The final prediction is made by combining the outputs of all weak learners through a weighted sum. GB also employs the concept of gradient descent to minimize the loss function, in comparison to RF. At every training step, a new weak learner is added to the ensemble and trained to predict the negative gradient of the model's current loss function [20]. Various loss functions can be utilized, but in this study MSE is used.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{3}$$

Here, n represents the number of samples, y_i is the true target value for the i-th sample, and $\hat{y}_i$ is the predicted target value for the i-th sample. The formula calculates the average of the squared differences between the true and predicted target values.

2.2.5 Neural network techniques

Neural Network (NN) techniques are a powerful class of algorithms, inspired by biological NNs. These NNs consist of interconnected layers of neurons that process and transmit information through weighted connections to other neurons. When trained, all of the weights and biases in the network are set to random values and training data is fed to the input layer. Data is passed through the network, getting multiplied and added together in different ways until it reaches the output layer. The weights and biases are adjusted until training data with identical features produce similar output. Because of the networks' ability to learn non-linear relations between features and labels, they are suited for complex problems [21].

The only NN technique used in this study is Multi-Layer Perceptron (MLP), which is the most utilized model in NN applications using the backpropagation training algorithm [22]. MLPs are standard NNs, and their architecture is decided by their hyperparameters. During training, the weights of the connections are adjusted using the backpropagation training algorithm. Backpropagation is a supervised learning method consisting of two main steps:

- **Foward pass:** As previously stated, input data is passed through the layers, multiplied by the weights, and passed through an activation function. This process is repeated throughout all of the network until the output layer is reached. By comparing the output to the true values, the error can be computed.
- **Backward pass:** The error is propagated back through the network, starting at the output layer and moving toward the input layer. While moving through the network, the gradients of the error with respect to each weight are computed using calculus. These gradients indicate how much the error would change if the weights were adjusted.

When the gradients are computed, the weights are updated using a learning rule such as gradient descent [23].

2.2.6 Instance-based learning

Instance-based learning, also called memory-based learning and lazy learning, is a family of machine learning algorithms. They make predictions based on similarities between input data and data in the training set. Instance-based learning algorithms keep the entire training set in memory and do not create a traditional model during the training phase. Instead, they learn when new data needs to be classified or predicted [24].

A popular example of instance-based learning is the k-Nearest Neighbors (KNN) algorithm, which is also used later in this study. When given input data, KNN searches the training data for the k instances that are most similar to the input according to a given distance metric. The labels of the instances are used to make a prediction on the input data. Rather than a loss function being explicitly optimized during training, the goal of training KNN algorithms is rather to fine-tune the two most important hyperparameters: k and the distance metric [25].

2.3 Concept Drift in Cloud Pricing Prediction

In the context of cloud pricing prediction, concept drift is an important consideration. Concept drift primarily refers to an online supervised learning scenario when the relation between the

input data and the target variable changes over time [26]. This suggests that a model trained to predict cloud pricing may fail to perform as intended if the market dynamic changes, even if the features used by the model remain the same.

It can be difficult to detect when concept drifts occurs. Using algorithms and tests such as ADaptive WINdowing (ADWIN) can therefore be crucial for addressing the issue of concept drift.

- **ADWIN:** A dynamic algorithm used to identify and adapt to changes in data streams. The core principle of the ADWIN algorithm is its use of a variable-sized window to maintain relevant statistical data. The window can expand or contract based on the characteristics of the data, allowing for an understanding of the evolution of data over time [27].

There are different types of concept drift, such as sudden drift, gradual drift, and recurring drift. Common techniques to address concept drift include:

- **Online Learning:** A technique where a model is being updated as new data becomes available. This allows the model to adapt to concept drift over time [28].
- **Active Learning:** An approach in which the learning algorithm may actively ask queries in the form of unlabeled instances to be labeled by an oracle. This can help the model to focus on the most informative samples for adapting to concept drift [29].
- **Ensemble methods:** Ensemble methods are learning algorithms that construct a set of classifiers and then classify new data points by taking a (weighted) vote of their prediction [20].

In this study, the potential of concept drift in cloud pricing prediction is acknowledged, and ways to reduce its effect are explored.

2.4 Related work

In this section, a review of relevant literature will be presented, with the intent to explain the state of research in the field of cloud pricing predictions. The literature was found using a Systematic Literature Review (SLR) methodology proposed by Kitchenham [30].

2.4.1 AWS PredSpot: Machine Learning for Predicting the Price of Spot Instances in AWS Cloud

A study titled ”AWS PredSpot: Machine Learning for Predicting the Price of Spot Instances in AWS Cloud” [2], was conducted in 2022. The study presents a machine learning approach to predict the prizes of AWS spot instances. Spot instances are a type of cloud service offered by AWS, where the users are given the option to bid for unused computing capacity. Their aim was to assist users in their bidding strategies.

The researchers used several machine learning algorithms, including linear regression, random forest, and gradient boosting, to find the most suitable approach for predicting spot instance prices. They evaluated the performance of the models using different types of performance metrics, such as Mean Absolute Error (MAE), MSE, and R-squared. Their results showed that

the gradient boosting algorithm was the best model in terms of prediction accuracy and generalizability.

2.4.2 Time-Series Analysis for Price Prediction of Opportunistic Cloud Computing Resources

A study titled ”Time-Series Analysis for Price Prediction of Opportunistic Cloud Computing Resources” [31], was conducted in 2022. This study uses the time series algorithm Seasonal Autoregressive Integrated Moving Average (SARIMA), to predict future cloud service costs. This algorithm is often used in different types of predictions, because of its ability to forecast complex data spaces containing cycles [32].

To evaluate the performance of the algorithms, the researchers used the performance metric MSE. The results showed that the algorithm was effective in predicting cloud service pricing.

While their study shows the potential of using time series algorithms, the research conducted in this thesis is more focused on the use of machine learning. Because of the limitation of data, the SARIMA algorithm was not effective.

2.4.3 A Novel Approach for Stock Price Prediction Using Gradient Boosting Machine with Feature Engineering (GBM-wFE)

In a comprehensive study [33] focused on stock prediction, a novel feature engineering approach was proposed for multiclass classification. The research utilized the WEKA library to explore the most effective feature selection algorithms and aimed to identify the optimal ensemble learning algorithm. The study primarily relied on stock data from the Nasdaq and S&P 500 indexes for the last 25 years, with an emphasis on monthly stock movement prediction.

A significant aspect of this research was the implementation of feature engineering to add two new features to the dataset: the mean value of the Open and Close price difference, and the High-Low difference. This enhancement significantly improved prediction accuracy, highlighting the potential of feature engineering in this field. An extensive range of algorithms was applied for feature selection, while various classifiers such as Stacking, AdaBoost, GBM, Multi-Boosting, and Random Forest were utilized for the Machine Learning aspect.

3 Technology

This chapter provides an in-depth exploration of the technological tools and platforms used in the research. Understanding these technologies is crucial to comprehend the methodology and results of this research.

3.1 Python

Python[1] is an adaptable programming language, known for its ease of use. It is often used in web development, data analysis, Artificial Intelligence (AI), and machine learning. Python's library and access to third-party packages, make it a popular choice.

In this research, there were several reasons to choose Python as the main programming language:

- **Readability and simplicity:** Python's clean and understandable syntax allows for fast development and tidy code.
- **Libraries and modules:** Python has access to a large number of libraries and modules for data processing, machine learning, and visualization. With libraries such as Pandas, NumPy, Scikit-learn, and Matplotlib, the implementation of the models is made simple.
- **Community:** Python is a popular programming language with a large community. Having access to documentation and tutorials makes it easier to find solutions to problems.

3.2 Visual Studio Code

Visual Studio Code[2] was used in the research. Because Python is supported by a wide range of Integrated Development Environment (IDE)'s, the IDE of choice was determined by personal preference. Visual Studio Code supports several features such as version control integration and customization that made it the preferred IDE.

3.3 Version Control

It is recognized that using version control in research is important for managing and tracking changes to code. The version controls used were Git and GitHub.

3.3.1 Git

Git[3] is a version control system that enables developers to manage and track changes in their code. There are several advantages of using Git:

- **Collaboration:** Git is ideal for efficient collaboration while producing code due to its features such as branching and merging.

[1] Python - https://www.python.org/
[2] Visual Studio Code - https://code.visualstudio.com/
[3] Git - https://git-scm.com/

- **Versioning system:** Git's versioning system track code changes, making it possible to revert to previous versions if necessary.
- **Integration:** Git has seamless integration with Visual Studio Code.

3.3.2 GitHub

GitHub[4] is a web-based platform that hosts Git repositories. It also offers some additional features such as code review and project management. There are several advantages of using GitHub:

- **Centralized repository:** Having a centralized repository allows for simple access for working on the code.
- **Code review:** GitHub's code review tools allow team members to easily analyze the code.
- **Interface:** GitHub's user-friendly interface simplifies the process of managing repositories.

3.4 Scikit-learn

Scikit-learn (sklearn)[5] is an open-source Python library. The library provides several tools for analyzing data and building machine-learning algorithms. The library's simple Application Programming Interface (API) and well-written documentation make it a popular choice. It was used in this research to build and test multiple price-predicting models.

3.5 Pandas

Pandas[6] is an open-source Python data manipulation and analysis toolkit. It provides tools for working with structured data. In this research, the Pandas DataFrame was important for working with tabular data with labeled rows and columns. The Pandas library can handle indexing, slicing, and reshaping of data as well.

3.6 Matplotlib

Matplotlib[7] is an open-source Python library. It is used to create visualisations of data and results. Matplotlib provides a variety of plotting functions and customization possibilities, making it an important tool.

In this research, Matplotlib was used for making clear and visual representations of the data, simplifying the work of analyzing and understanding the results. Furthermore, the library was used to compare machine learning algorithms and tune parameters. This enabled effective communication of findings and evaluation of the approach's efficacy.

[4]GitHub - https://github.com/
[5]sklearn - https://scikit-learn.org/stable/
[6]Pandas - https://pandas.pydata.org/
[7]Matplotlib - https://matplotlib.org/

3.7 Training and testing environment

Training and testing of the models were done on one system, the personal computer of one team member. The technical specifications of the system are listed below.

Type	Model
CPU	Intel i7-12700KF
GPU	NVIDIA GeForce RTX 3080
RAM	Kingston 32 GB
SSD	Seagate FireCuda 530 2TB

Table 3: Technical specifications of the system used for training and testing.

4 Method

Research methods are systematic approaches, techniques, and tools used to collect, analyze, and interpret data. The goals of these methods are ultimately generating knowledge, testing hypotheses, and answering research questions. They are essential for reliable and valid scientific research, which leads to the advancement of knowledge. The use of well-established research methods ensures that research findings are trustworthy and reproducible. This objectivity allows researchers to compare their findings with others and build on previous work, identifying trends and refining existing knowledge.

Research methods also provide a structured framework for conducting research, which helps researchers organize their work systematically and logically. They facilitate a comprehensive understanding of the research process and its outcomes, contributing to the growth of knowledge. Additionally, research methods ensure that findings are generalizable and transferable to other contexts or populations, expanding the applicability of the knowledge generated.

By following established guidelines or principles, research methods also address ethical considerations, respecting the rights and welfare of the participants and stakeholders involved. This maintains the integrity of the research and the knowledge it generates, making research methods indispensable for advancing knowledge in various fields and informing evidence-based decision-making [34].

4.1 Systematic Literature Review

In this research, a method proposed by Kitchenham for conducting a SLR was adopted. The aim of the SLR was to provide an unbiased evaluation of existing literature on cloud predictions. For the purpose of this research, the process has been divided into three main stages.

4.1.1 Planning the Review

It is important to develop a protocol for the review, define the research questions and determine the strategy for searching and selecting literature. The protocol should act as a guideline throughout the review process to ensure consistency and reduce potential bias.

The review is guided by the following questions:

- What are the common methodologies for predicting cloud service costs?
- What machine learning models have been previously applied to cloud cost prediction?
- How have previous studies approached model selection and evaluation for cloud cost prediction?
- How has the SARIMA algorithm been used for cloud price prediction?

As the primary source of literature, the digital database: Google Scholar was used. Search terms used were combinations of ”cloud computing”, ”price prediction”, ”machine learning”, ”time series”, and other related terms.

4.1.2 Conducting the Review

After establishing the protocol, the reviewing process was initiated. This involved executing the search strategy, selecting appropriate studies for review, and evaluating relevant data.

Study Selection: A rough evaluation was first conducted, removing irrelevant studies based on their title and abstract. The remaining studies were then evaluated and chosen based on the full text.

Data Extraction: From the selected studies the information regarding the models, the type of data, and the methodologies were extracted.

4.1.3 Reporting the Review

In the final stage, a quality assessment of the selected studies was conducted, and the data was evaluated to answer the research questions.

Quality Assessment: A set of predefined criteria to evaluate the reliability and validity of the findings was set. These were criteria such as:

- Was the methodology adequately described?
- Was the data used in the study clearly described?
- Were the statistical methods used to analyze the data appropriately?
- Were the results clearly presented?
- Was there a clear statement of findings?

Data Evaluation: The data was then evaluated, making it possible to draw conclusions about the state of the research field and answer the research questions.

The findings of this SLR form the basis of Section 2.4 in this study, enabling an understanding of the landscape of cloud pricing prediction research.

4.2 Design Science Research

Design Science Research (DSR) is a systematic and iterative approach to problem-solving that focuses on creating and evaluating innovative artifacts to address real-world issues. The process begins with identifying a research problem motivated by a practical concern or a gap in existing knowledge. Next, the objectives of the solution are defined, specifying desired outcomes and performance criteria.

An artifact is then developed. The artifact can be a method or a tool, or in this case, a model. It aims to address the problem and achieve the defined objectives. Later, the artifact is evaluated using appropriate methods, such as simulations, experiments, or case studies, to demonstrate its utility, quality, and impact. The results of the DSR process, including the problem, artifact, and evaluation findings, are communicated through documentation.

Finally, the process involves reflection and iteration, during which researchers consider the findings, identify potential improvements, and refine the artifact while deepening their understanding of the problem domain. This iterative and structured approach allows for the development and evaluation of innovative solutions that contribute to both theoretical knowledge and practical applications [35].

4.2.1 Process

DSR was used in this study to evaluate the work, as well as to structure the iterative progression of the presented approaches. The problem domain being explored was defined by SkyeTec AS, but the research objectives were defined by the team.

The two proposed approaches were developed iteratively. After spending time getting to know the dataset and reviewing relevant literature on ML techniques and cloud services, work began on a monolithic approach. The model type was selected and the hyperparameter tuning was automated. As the work on the monolithic approach concluded, the team consulted their supervisor for evaluation and reflection on the work. It was then decided to try a different, multi-model approach to the problem. The multi-model approach would be a cluster of models, each handling predictions of its own type. Although similar to the monolithic approach, this would allow for each of the models to be individually defined with respect to model type and hyperparameters, which possibly could boost performance.

After development, the approaches were evaluated and reflected upon, both individually and in comparison to one another. As the final step of the DSR methodology, the entire process is documented.

In this study, the team has successfully employed the DSR methodology, resulting in the development and evaluation of innovative artifacts that address the research question. The DSR approach provided the team with a structured and rigorous framework for creating a practical and theoretically grounded solution. Utilizing DSR was found to be rewarding, as it allowed the team to contribute meaningful insights to the domain.

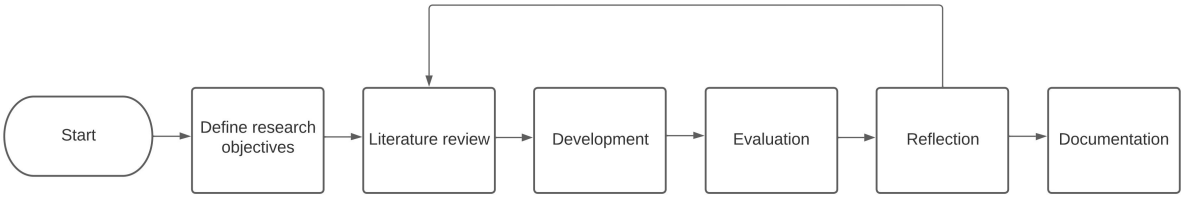


Figure 1: Flowchart of the DSR-based work.

4.2.2 Roles

The team consists of two members who have developed a strong working relationship through participating in several group projects and living together. Both members are able to initiate,

organize and complete tasks, both independently or in collaboration. It was therefore established that there was no need for special assigned roles, other than distributing workload and assigning a secretary.

4.3 Data collection

The dataset used to train the models was acquired from SkyeTec AS and contains data points with the logical shape shown in the listing below. It describes the Azure IaaS and PaaS resources used by a stable customer over 2022 and contains over 900000 samples of resource usage, billed by both PAYG and RIs.

```
{
    "Cost": 1.5
    "CostUSD" : 0.17,
    "ResourceId": "/subscriptions/-----------------"
    "ResourceType" : "microsoft.storage/storageaccounts"
    "ResourceLocation": "eu west"
    "ResourceGroupName" : "cloud-shell-storage-westeurope",
    "PublisherType" : "Microsoft"
    "ServiceName": "Storage"
    "Meter" : "All Other Operations",
    "tags": [
          []
       ],
    "Currency" : "NOK"
}
```

Figure 2: JSON version of a row in the dataset.

4.3.1 Data preprocssing

Before training the models the data has to be preprocessed with the goal of making it usable for training.

- **Data cleaning:** Missing values are searched for in each Pandas DataFrame throughout the data cleaning process. This is done by iterating over the list of DataFrames and printing the count of null values in each. This is to identify any columns or rows that require further attention or handling. This can include filling in missing data or dropping incomplete data.

  After checking for missing values, the DataFrames are combined into one single DataFrame, using the ”pd.concat()” function. By setting the ”ignore_index” parameter to ”True”, it is ensured that the index values are reset. This step is important for consolidating the data and preparing it for further processing and analysis.

- **Data Transformation:** In the data transformation process, the ”pd.get_dummies()” function is applied to the combined DataFrame. This function converts categorical variables into dummy variables, which are binary columns that represent the presence or absence of a category in the original variable. Doing this makes it easier for the machine learning algorithm to understand the correlation between features.
- **Feature engineering:** In the feature engineering process, dummy variables for categorical features are created. The 20 most frequent categories for each categorical column in the dataset are chosen to prevent the data from becoming too large. To accomplish this, a custom function is used. This function takes the original dataset as input and returns a transformed dataset with the top 20 most frequent categories for each categorical column. The remaining categories are grouped as ”Other”. This approach reduces dimensionality while retaining the most relevant information from the categorical features. It can also improve the performance of the machine-learning models by limiting the number of categories used.
- **Data splitting:** Data splitting is a useful step in the machine learning process, as it makes it possible to evaluate the performance of the models on unseen data. In this analysis, sklearn’s ”train-test-split” function is used to divide the dataset into training and testing sets. The data is split into training sets containing 70% of the data, and the test set containing the remaining 30%. The ”random_state” parameter is set to 42, to ensure consistent splitting across multiple runs.

  By splitting the data and evaluating several models on the test set, it is possible to assess their generalization performance and compare them. This ensures the selection of the best model for this use case.
- **Concept drift points:** Recognizing the concept drift points in the data can be important when evaluating the data, as this can help understand where these drifts occur and how they might influence the accuracy of the models. In this study, an attempt was made to detect these drift points using the ADWIN algorithm with a delta of 0.01.

4.4 Model selection

Model selection is one of the most crucial aspects of a successful implementation. Using the works discussed in section 2.4, a pool of models to be examined and tested was chosen. All of the models are supported by sklearn, a massive time saver with respect to model building, training, and testing. The models were built with sklearn to make sure they are easily reproducible and optimized.

Model name	Keywords
Random Forest	Ensemble learning, decision trees
k-Nearest Neighbors	Instance-based learning, memory-based
Multi-layer Perceptron	Neural network, backpropagation, gradient descent
Ridge	Linear, L2 regularization
Gradient Boosting	Ensemble learning, weak learners, sequential

Table 4: Model pool with keywords.

4.4.1 Monolithic approach

For the monolithic approach, the RF model was chosen after testing and evaluating all the models in the model pool. By reviewing [36], the hyperparameter domains were set, and by using sklearn's RandomizedSearchCV, the hyperparameters were fine-tuned. This function randomly searches the provided domain to find the best-performing hyperparameters.

Resource Type	estimators	min_samples_split	max_nodes	max_features	max_depth
RF	1000	50	9	auto	6

Table 5: Hyperparameters of the RF model.

4.4.2 Multi-model approach

Model selection and hyperparameter tuning for the multi-model approach is more complex. Hyperparameter domains were set by reviewing sklearn's model documentation, and models and hyperparameters were selected by looping over the resource types and fitting all the models in the model pool to the given subset of the data using RandomizedSearchCV. This way all the combinations of models and resource types can be evaluated fairly, using something close to their best hyperparameters from the predefined domain. When selecting models and hyperparameters, R^2 was favored over MSE.

Resource Type	subsample	estimators	min_samples_split	min_leaf	learning_rate
Express Routes	0.8	1000	250	20	0.05
VM network	0.5	500	500	20	0.1
Disks	1	1000	1000	50	1
Key vaults	1	100	250	10	0.05
Bastion	0.8	100	1000	10	0.05
Namespaces	1	1000	250	10	0.05
VMs	1	100	250	50	0.05
Pricings	0.8	200	500	20	0.05
Firewalls	1	100	1000	20	0.05
Public IP	0.5	1000	500	10	0.05
Containers	0.5	500	250	20	0.5
Galleries	1	1000	500	10	0.1
Snapshots	0.5	500	250	50	1
Connections	0.5	500	500	10	1
Network watchers	1	100	250	10	0.05
Restore points	1	1000	250	20	1
Metric alerts	0.5	100	250	10	1

Table 6: Hyperparameters of the GB models.

Resource Type	estimators	min_samples_split	max_nodes	max_features	max_depth
Load balancers	750	75	3	auto	6
VM scls	750	50	9	auto	3
Registries	250	75	3	sqrt	3
Private endpoints	750	100	3	auto	6
Action groups	250	100	9	auto	3

Table 7: Hyperparameters of the RF models.

Resource Type	activation	alpha	hidden_layer_sizes	learning_rate	solver
Recovery valuts	tanh	0.498045	125	constant	sgd

Table 8: Hyperparameters of the MLP model.

Resource Type	alpha
Workspaces	3
Images	1
Private DNS	3
Scheduled query rules	0
Storage	0.5

Table 9: Hyperparameters of the ridge models.

The KNN model did not end up being the best-performing model on any of the resource groups and is therefore not represented by a table with hyperparameters.

4.5 Model evaluation

Model evaluation is important to understand whether or not a trained machine learning model is performing well. It is possible to do so by using the model to predict, and then compare the accuracy of the results to the actual data that the model is attempting to predict. There are several ways to evaluate the model's accuracy. In this study, performance metrics, cross-validation, and testing were used.

4.5.1 Performance metrics

Performance metrics are used to understand how the machine learning model is performing. By evaluating the numbers given by performance metrics, it is possible to evaluate whether the model is progressing. The evaluation metrics used in this research are MSE, R^2, and a differentiation algorithm used to compare how far the overall prediction is from the actual data percentage-wise. To compare the approaches, R^2 and the differentiation algorithm are calculated using the sum of predicted prices for each day and the actual sum of prices.

4.5.2 Testing and Cross-validation

To understand how a machine learning model is performing, it is important to test its predictions. This involves using separate data for training and testing. In this research, the models were trained on 70% of the dataset and tested on 30% of it. This is crucial to ensure that the model can predict unseen data.

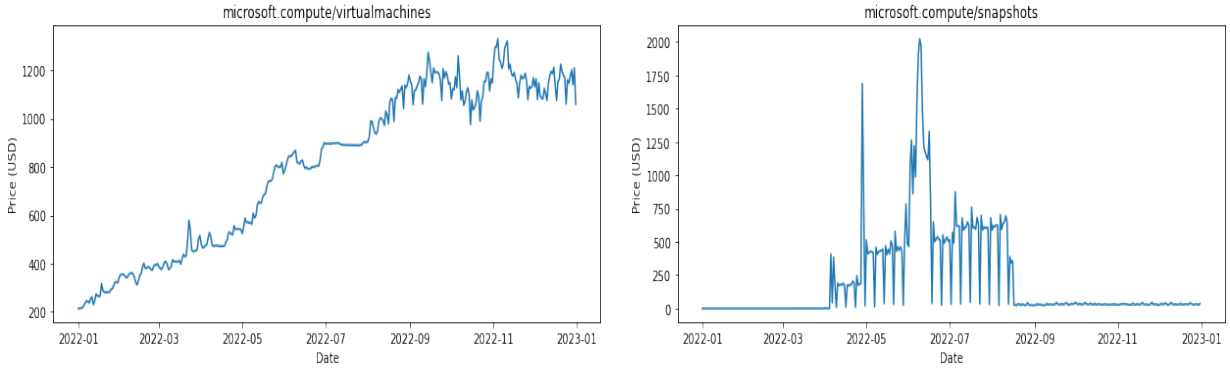
Cross-validation is an important measure to further validate the model's predictions. In this research, sklearn's KFold and split methods were used to split and test the data. The KFold configuration used included 5 splits, a shuffle, and a random state of 42. The 5 splits specify that the data should be divided into 5 equal folds, with each fold being used as a test set once. Setting the shuffle parameter as true enables shuffling of the data before it is split into folds. This helps ensure the integrity of the data distribution across all folds. Using 42 as a random state ensures that the same random seed is chosen each time, making it easier to reproduce the results.

5 Results

The results chapter presents the most important findings derived from applying the methodologies described in the previous chapter. It provides a clear presentation of the outcomes using tables and data plots. By presenting the results, this chapter lays the foundation for the discussion and interpretation of these findings in relation to the aims of the research. See table 1 and 2 for complete lists of the short forms used in these visualizations.

5.1 Scientific results

5.1.1 Dataset



(a) The daily cost of virtual machines.

(b) Daily cost of snapshots over the same period.

Figure 3: Comparison of Virtual Machines and Snapshots Costs.

The comparison in figure 3 shows the daily cost trend of different resource types over a year. Different resource groups have varying use throughout the year, for example, snapshots are almost exclusively used in the summer.

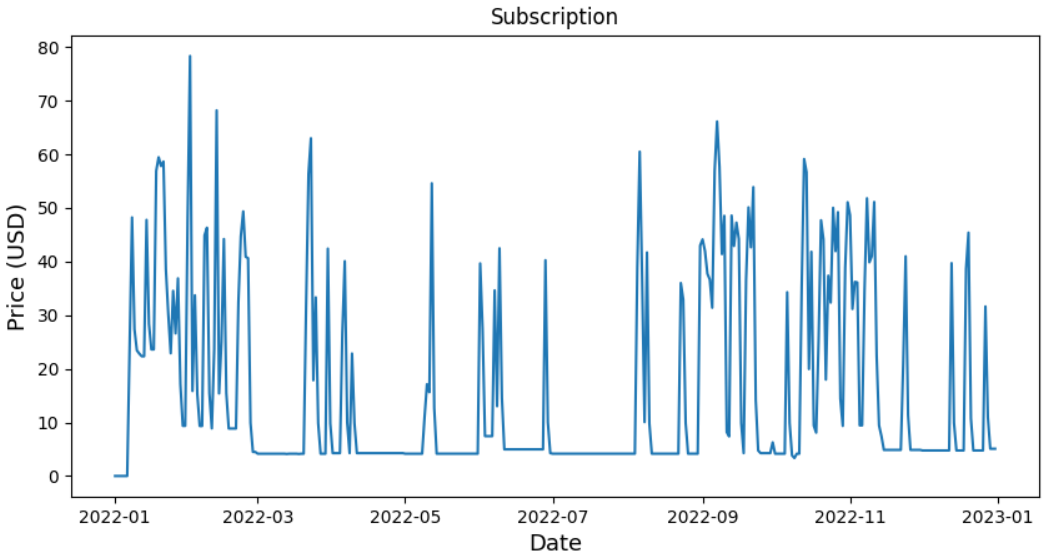


Figure 4: Daily cost trend of a subscription.

Similarly to figure 3, figure 4 shows a daily cost trend. However, this cost trend is of a subscription, meaning it describes a number of resource groups, which in turn contain resources.

5.1.2 Monolithic approach

	Model	MSE	R^2	Diff%
Monolith	RF	5.23440	0.39409	0.83527

Table 10: Performance of the monolithic model.

In the tables describing model performance Diff% is a metric of the percentage-wise deviation of the average prediction compared to the actual price.

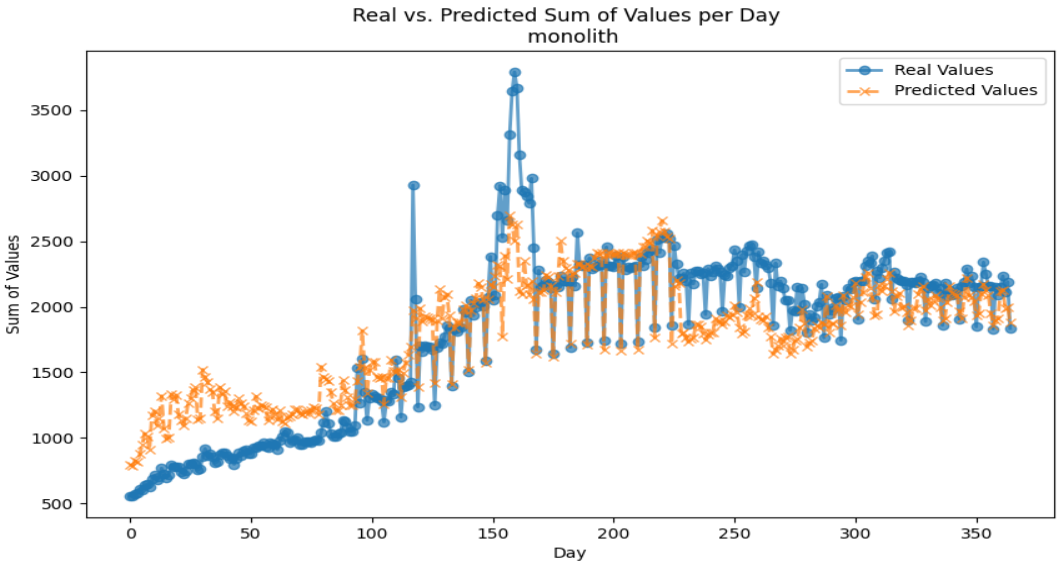


Figure 5: Comparing the monolithic model's predicted values with the actual prices.

Using cross-validation, the monolithic model predicts the values of the entire dataset. All resource usage from each day is summed and compared to the sum of the predicted prices, provided by the model. The blue circles represent the sum of one day's prices and the yellow crosses represent the sum of the predicted prices of one day. This will be the case for all similar plots in this chapter.

5.1.3 Multi-model approach

ResourceType	Model	MSE	R^2	Diff%
Action groups	RF	2.8789e-08	0.08847	2.08337
Bastion	MLP	0.29020	0.94847	0.49575
Connections	MLP	9.6882	0.15341	4.19723
Containers	MLP	0.39875	0.51852	1.55628
Disks	GBR	0.01683	0.97105	0.30032
Express Routes	MLP	0.00103	0.99856	0.31868
Firewalls	RF	57.5296	0.64337	1.06818
Galleries	RF	6.9400e-05	0.35453	0.97092
Images	KNN	6.3554e-05	0.77899	8.80442
Key vaults	RF	4.6909e-07	0.96835	1.37820
Load balancers	GBR	0.00115	0.98617	1.03174
Metric alerts	GBR	1.7661e-06	0.04192	0.14400
Namespaces	GBR	0.00105	0.93693	1.35363
Network watchers	GBR	0.00434	0.13968	4.26526
Pricings	RF	0.04856	0.67062	6.80992
Private DNS	Ridge	3.3628e-06	0.61657	0.81430
Private endpoints	Ridge	0.00036	0.35356	0.50742
Public IP	GBR	6.3578e-05	0.61336	0.10208
Recovery valuts	RF	12.1249	0.81994	0.77530
Registries	Ridge	0.00028	0.85496	1.90799
Restore points	GBR	1.3258	0.09811	11.90125
Scheduled query rules	Ridge	5.3098e-05	0.76390	0.25654
Snapshots	GBR	5.8764	0.20436	4.28823
Storage	GBR	2.7159	0.69901	1.15062
VM network	RF	0.71532	0.97567	0.25741
VM scls	RF	0.09427	0.96232	1.64063
VMs	GBR	4.2927	0.68227	0.67542
Workspaces	Ridge	7.5028	0.22986	2.61872

Table 11: Performance of the models in the multi-model approach.

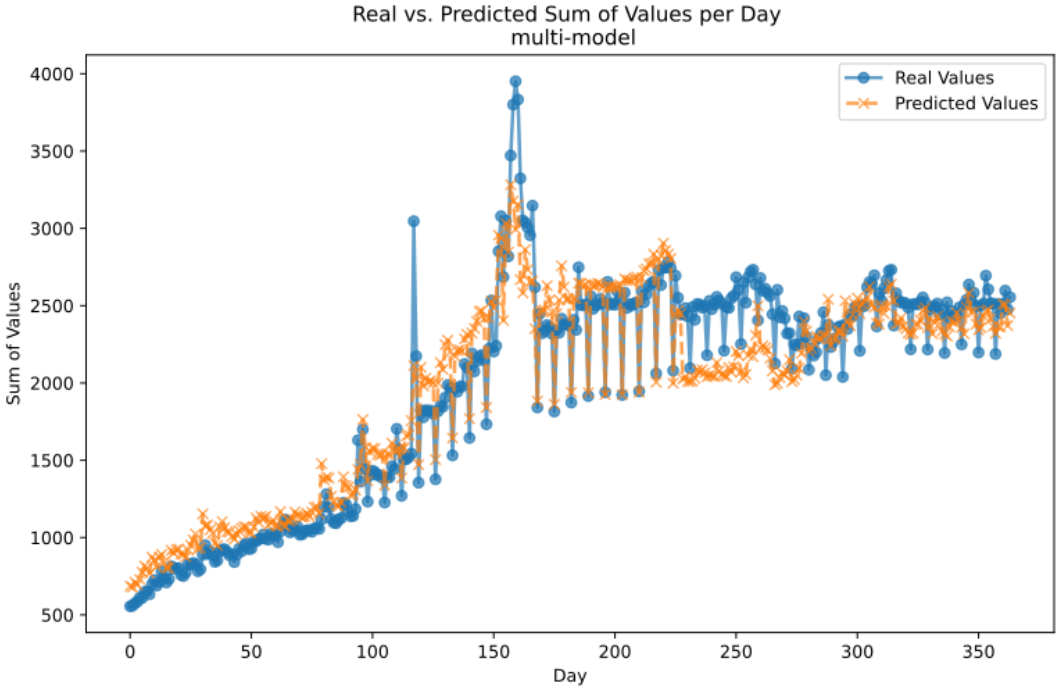


Figure 6: Comparing the predicted values of the multi-model approach with the actual prices.

Similarly to figure 5, the model cluster of the multi-model approach predicts the prices of the resources in the dataset using cross-validation in figure 6.

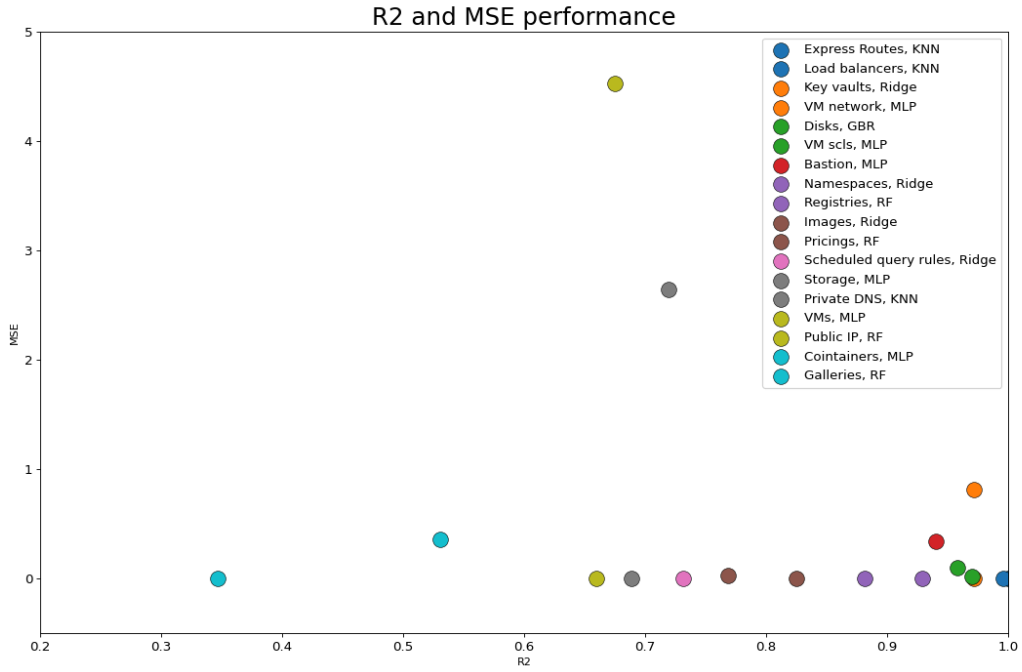
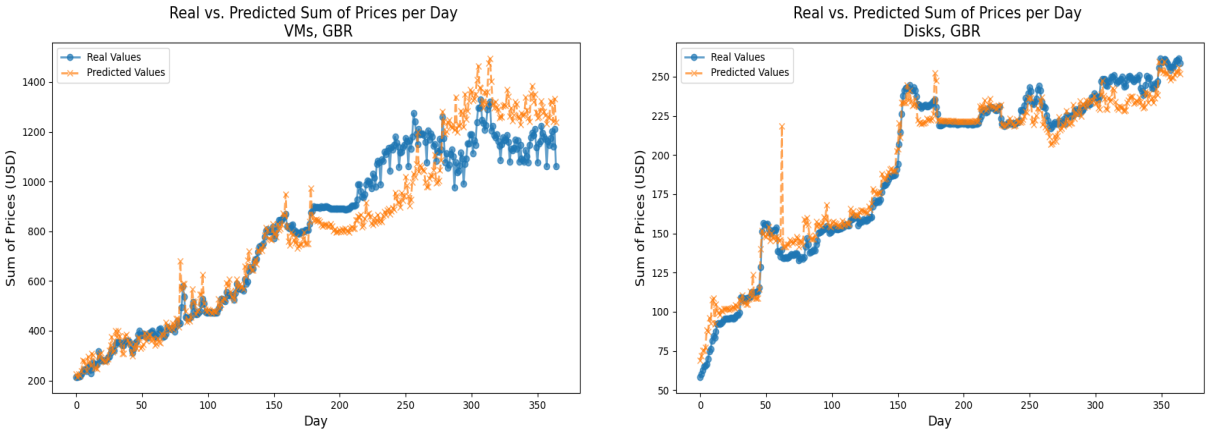
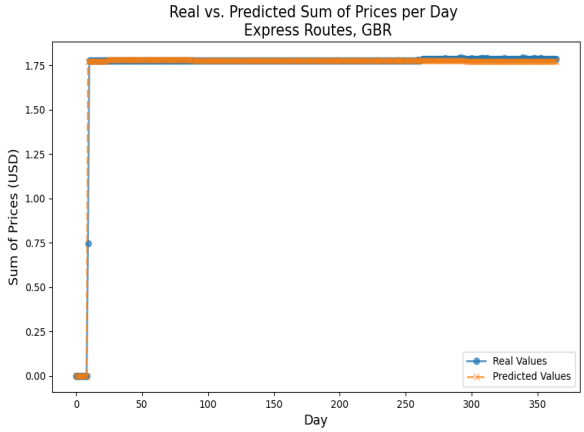


Figure 7: Visualization of select models within the multi-model approach.

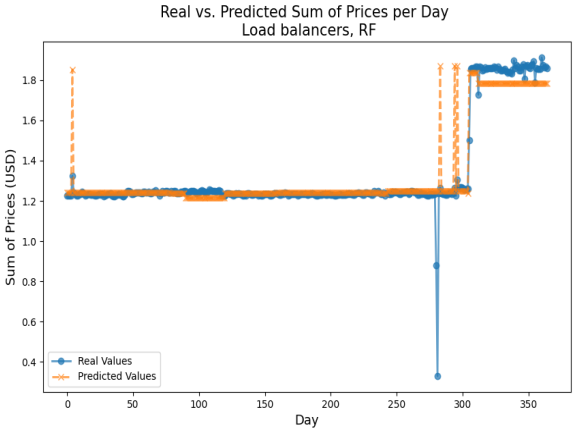


(a) Predicted VM prices versus actual VM prices. (b) Predicted disk prices versus actual disk prices.

Figure 8: Predictions from the models of the two most expensive resource types.

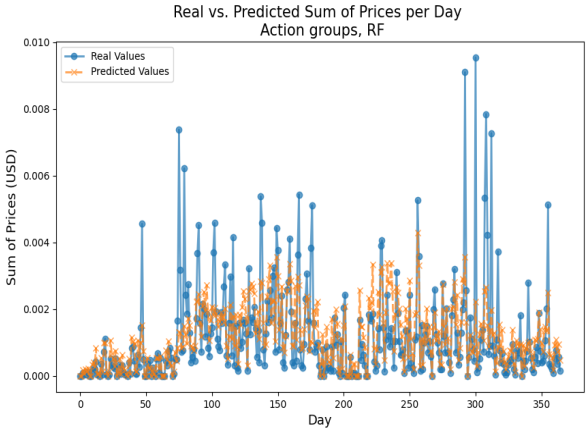


(a) Express routes.

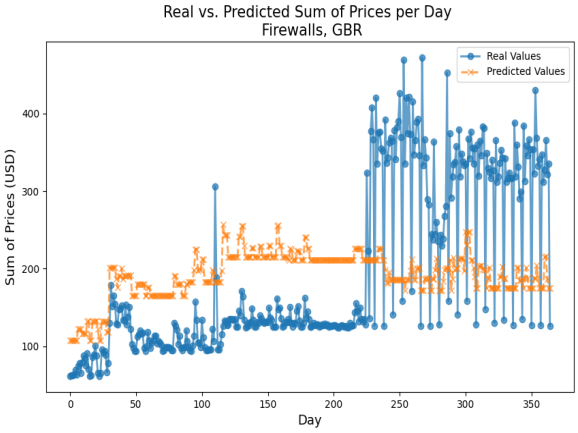


(b) Load balancers.

Figure 9: Predictions of some of the best-performing models in the multi-model approach.

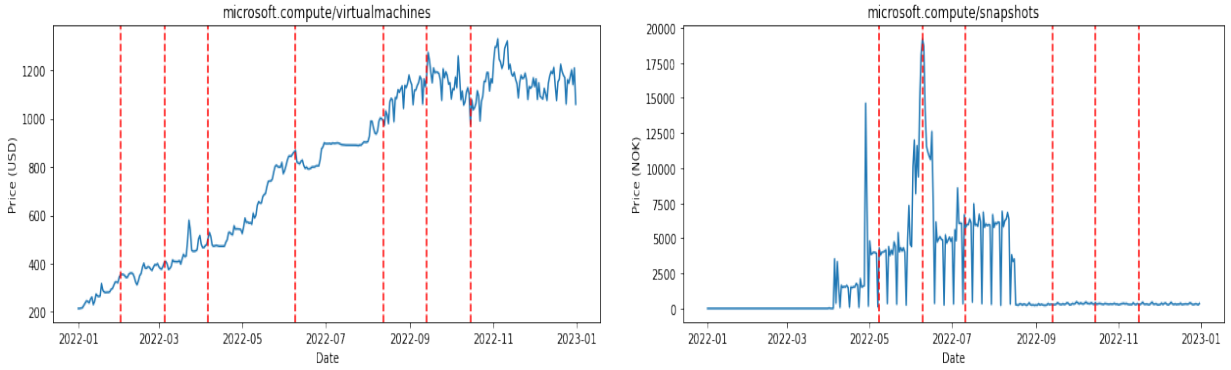


(a) Action groups.



(b) Firewalls.

Figure 10: Predictions of some of the worst-performing models in the multi-model approach.



(a) Virtual Machines.

(b) Snapshots.

Figure 11: Comparison of ADWIN Algorithm's Virtual Machines and Snapshots Drift Points.

5.1.4 Comparison of the approaches

Approach	R^2	Diff%
Monolith	0.82124	0.08420
Multi-model	0.90121	0.29754

Table 12: Performance metrics from the comparison of the approaches.

5.2 Administrative results

5.2.1 Project Management and Process

The initial plan was to use the Scrum framework for project management, dividing the work period into 5 sprints as seen in attachment 1b. However, it quickly became apparent that the DSR methodology would be more suitable for the specific problem given by SkyeTec AS. The DSR methodology offered a more systematic approach to problem-solving, focusing on creating and evaluating price-prediction models. The research problem was initially identified, and solution objectives were defined based on this. As the project progressed, it was discovered that there was a limited amount of relevant research available, which led to the decision to refine the objectives and conduct independent research instead.

DSR helped guide the project through an initial SLR and a second review after updating the research objectives. Several approaches were explored, creating two solutions: a monolithic approach and a multi-model approach. These approaches were developed with consultation from the project supervisor and underwent evaluation and reflection both individually and in comparison to one another.

5.2.2 Time Allocation and Development

Time was allocated between research, development, and documentation throughout the project. Through planning and meetings, it was ensured that the focus was on the most critical aspects of the project. The iterative nature of DSR allowed for the continual understanding of the problem, and improvement of the solutions.

The transition from Scrum to the DSR approach preserved an organized framework for research and development. This approach proved to be rewarding, as the team contributed insights to the area, while also addressing the problem presented by SkyeTec AS.

In conclusion, the administrative results demonstrate the team's ability to adapt and manage a project using the DSR method. The choice of approach provided a structured and effective process for creating and evaluating innovative machine learning models, resulting in a successful project that met the objectives.

6 Discussion

The discussion chapter provides a comprehensive discussion of the findings presented in the results chapter.

6.1 Data

While analyzing and processing the data, the decision was made to have the models predict resource prices. Up until this point, the plan was to have the models predict the evolution of the resources within subscriptions, and the total prices of these. It became apparent that the data was insufficient for this type of modeling, as the number of data points was lowered to $1/30$ of the original. The task of modeling the number of resources over time is an interesting and relevant aspect of predicting cloud prices, but not something the team can provide meaningful insight on with the current data.

It is the team's assessment that centering the models around resources instead of subscriptions is correct. As seen in figure 4, a subscription behaves in a way that is hard to predict, especially given the relatively low amount of data the dataset provides when augmented to describe subscriptions.

The dataset used in this study is limited, and one of the most significant constraints of the work as a whole. It was collected from a single customer over a one-year period. While this allows for an in-depth examination of the specific customer's usage, it is essential to recognize the limitations that come with using this type of dataset. Patterns and trends observed in the data could be unique to the single customer in question, and may not reflect the behavior of other customers or the overall market.

Efforts were made to enhance the dataset by using AWS' publicly available virtual machine instance prices. Because the training data is not labeled with the operating system, this would not contribute to the model's performance, and the enhancement was ultimately dropped. If the operating systems of virtual machine instances become a part of data extracted from

Azure, AWS's prices could provide more depth to the dataset and make both the monolithic and the multi-model approach better at predicting prices of VMs.

Because of the relatively low amount of computing power needed to train the models, it could be beneficial to use these approaches to train personalized price predictors for individual customers.

6.2 Models

In this research, two approaches for machine learning models were developed and tested - the monolithic and the multi-model approach. The choice of these two approaches was influenced by the problem presented by SkyeTec AS and the data available.

The monolithic model treats the data as a single entity and is able to identify global patterns and trends. In table 10 and figure 10, we can see the model performing well. However, due to the diversity of the data, it is likely that this approach may fail to identify more local or resource-type-specific trends.

Because of this, the multi-model approach was developed on the understanding that different resource types would have unique patterns and trends. By creating individual models for each resource type, this approach aims to capture these trends more accurately. In table 11 and figure 6 to 10, the performance of the multi-model approach is presented and visualized. The limitations of this approach lie in its requirements of maintaining and managing multiple models, and the limitation of data for some resource types.

As seen in table 12, the multi-model approach has a higher R^2-score than the monolithic approach, indicating it does a better job of explaining the variance, predicting patterns, and identifying intricate relations in the data. However, the monolithic approach has a lower, and better, average deviation percentage despite its lower R^2-score. This could mean that the monolithic approach produces more precise predictions in terms of accuracy. One interpretation of these results might be that the multi-model approach could be overfitting, even though it offers a more nuanced and detailed understanding of the dataset. On the other hand, the monolithic model can be underfitting, and unable to capture the data's complexity. Still, it offers more precise and consistent predictions.

Which one is better depends solely on the preferred nature of the predictions. If a model that understands intricate patterns and can predict seemingly complex price changes is desired, the multi-model is the best choice. If a model that is stable and reliable is desired, the monolithic approach will suit the task better.

A constraint of the work is the computing power and hardware that we had available during the development of the approaches. For example, by capping the number of features considered by the fitting algorithm. Without taking this countermeasure, the dummy-encoded dataset is simply too large for the memory of the testing environment, even when dropping the "tags"-column which proved to not provide anything to the models. During the fitting of the final models, the number of features was set to 30. Additionally, the number of iterations in the hyperparameter search was set to 10, also because of hardware limitations. It would naturally be optimal not to have these constraints, and though the implications of them likely are small, it is important to acknowledge them.

While evaluating the implemented models, R^2 was used as the primary performance metric. It was seen as the most important because it describes how well the model explains the variance in the dataset. Given the results, this was an appropriate assumption as the models with higher R^2 outperform models with lower R^2 with respect to the model accuracy, regardless of MSE. When comparing the two approaches, MSE also becomes less interesting. As the approaches are compared by summing all the predictions for each day, the MSE becomes inflated and provides negligible insight into the models' performance.

6.3 Concept Drift Points

The process of identifying concept drift points in the data is an important factor to understand the fluctuations that could affect the accuracy of the machine learning models over time. The ADWIN algorithm was used in this study to detect these drift points. However, the effectiveness of this algorithm varied depending on what data it was applied to.

The algorithm was relatively successful when applied to the resource type ”virtual machines”, showing some capability in detecting concept drift points. However, when applied to the resource type ”snapshots” it was not as effective.

The variation of the success of the ADWIN algorithm may be a result of differences in data patterns across the different resource types. This might suggest that the algorithm's parameters need fine-tuning to improve its performance across all resource types. This can be a difficult task, as the concept drift points isn't pre-defined in the data.

6.4 Process

Initially, the objectives were to reproduce established research and thus create a price estimation tool for SkyeTec AS' customers. Through literature review, the team found the amount of publicly available relevant research to be too low for the objectives at the time, studies using the same kind of data the team acquired were non-existent. Therefore, the team decided to conduct the research themselves, and the research objectives were changed to exploring different approaches to ML for cloud price prediction and to provide the developers at SkyeTec with a study to base their own research on.

With the research objectives in mind, a literature review was conducted. The most relevant studies were on price prediction of VMs, typically within the AWS ecosystem. AWS allows its users to access historical pricing data of their VM instances, something Azure does not. Studies regarding AWS have the cloud service provider as a massive data source, allowing for models like Autoregressive Integrated Moving Average (ARIMA) and SARIMA to be employed. Unfortunately, models like these require comprehensive data over a long period of time, which is not made available by Azure. Microsoft recently deployed a public API where current retail prices can be fetched. In the long term, this API might offer the functionality that AWS' API does, but until then users' own data has to be used.

6.5 Comparison with Previous Studies

In section 2.4.1, a study on predicting cloud spot prices is presented. It used several machine learning models to predict spot prices in the AWS cloud. The research relied on a large historical price dataset from AWS for model training and testing. In contrast, this study relies on a year's worth of data, collected from one customer.

This research was also split into two modeling approaches: a monolithic model that treated all costs as a collective entity, and a multi-model method that used one model for each distinctive resource type. Such a strategy would not be suitable for the problem provided in the AWS PredSpot article.

In section 2.4.2, another study using time-series analysis for cloud price prediction is presented. It proposes the use of the SARIMA algorithm for prediction. While the SARIMA algorithm is proven successful in some contexts, the research conducted in this thesis found it unusable due to the limited dataset. This highlights the importance of model selection and evaluation in relation to the data available.

In section 2.4.3, a study that focused on stock prediction is presented. It utilized the WEKA library to explore the most effective feature selection algorithms and aimed to identify the optimal ensemble learning algorithm. While there are similarities in the methods used and the goal of using machine learning for multivariable price prediction between the two studies, the domain, and the approaches are inherently different.

Although this research aligns with previous studies in the general use of machine learning models for cost prediction, it also presents unique approaches due to the nature of the dataset.

6.6 Implications of the Research

The findings of this study have several implications for both the research and the practical areas of cloud service cost prediction.

In terms of research, this study presents some of the potential and limitations of using machine learning models to cloud cost prediction with a customer-specific dataset. The research introduces a unique approach by using both a monolithic and a multi-model and comparing these strategies.

From a practical perspective, this study can help guide businesses in their cost management efforts. The findings suggest that it is possible to make meaningful predictions about future cloud service costs using machine learning, even with a limited amount of data. This study also highlights the potential benefits of tailoring prediction models to specific resource types in the cloud environment, enabling more specific predictions.

However, recognizing the limitations of this study is important. The data used in this research was collected from a single customer over a one-year period. The applicability of this method to all businesses or broader market trends is therefore uncertain. Further research with a larger and more diverse dataset would be needed to validate and generalize the results further.

7 Conclusion

This study presents two machine learning approaches for predicting Azure cloud cost: a monolithic model and a multi-model. In the research, our findings suggest that the approaches have different strengths and weaknesses. The monolithic model seems to be more accurate and stable with its predictions, while the multi-model approach might be better at identifying underlying patterns, and accounting for the variance in the data. While both approaches have their benefits and limitations, the findings of this research can contribute to the field of client-tailored cloud cost prediction. It can also support better decision-making for companies using Azure cloud services.

Our findings indicate that machine learning can be an adequate tool for predicting cloud costs for Azure services using both presented approaches. The two approaches average daily price predictions under 0.3% from the actual prices and have an R^2-score over 0.8 using one of SkyeTec AS' customer's data.

7.1 Future work

While this study has provided valuable insights into Azure cloud cost prediction, there are several areas for further research and development.

SkyeTec AS wants to use cloud cost prediction to help customers with better expense prediction and understanding. While the models in this research are an important start, making the models available for the customers will be crucial.

The models developed and evaluated in this research present an important step toward assisting customers with improved expense prediction and understanding. However, to make them more customer-tailored and future-proof, further investigation and improvements are advised.

- **Concept drift:** One aspect that could benefit from more exploration is the handling and understanding of concept drift. By making the models more adaptable to changes in data, the accuracy and reliability of the predictions can be enhanced.
- **Active learning:** The current models are trained and evaluated on a static dataset. By using an active learning approach, the performance of the models could continually be improved, making the implementation more future-proofed.
- **Testing on different customers:** To ensure generalizability and applicability, the models should be tested on different customers' data. The models and hyperparameters used in this research are specialized for one specific customer. Testing and evaluation for other customers are therefore important to uphold the accuracy of the models.

References

[1] F. Richter. "Infographic: Big three dominate the global cloud market", Statista Infographics. (Apr. 28, 2023), [Online]. Available: https://www.statista.com/chart/18819/worldwide-market-share-of-leading-cloud-infrastructure-service-providers (visited on 05/14/2023).

[2] A. Baldominos, Y. Sáez, D. Quintana, and P. Isasi, "AWS PredSpot: Machine learning for predicting the price of spot instances in AWS cloud", *International Journal of Interactive Multimedia and Artificial Intelligence*, vol. 7, pp. 65–74, Feb. 8, 2022. DOI: 10.9781/ijimai.2022.02.003.

[3] S. Brown. "Machine learning, explained", MIT Sloan. (Apr. 26, 2023), [Online]. Available: https://mitsloan.mit.edu/ideas-made-to-matter/machine-learning-explained (visited on 05/01/2023).

[4] "Om Oss", SkyeTec AS. (), [Online]. Available: https://skyetec.no/omoss/ (visited on 05/15/2023).

[5] T. Chen, T.-T. Chuang, and K. Nakatani, "The perceived business benefit of cloud computing: An exploratory study", *Journal of International Technology and Information Management*, vol. 25, no. 4, Jan. 1, 2016, ISSN: 1941-6679. DOI: 10.58729/1941-6679.1297. [Online]. Available: https://scholarworks.lib.csusb.edu/jitim/vol25/iss4/7 (visited on 04/18/2023).

[6] "Advantages of cloud computing", Google Cloud. (), [Online]. Available: https://cloud.google.com/learn/advantages-of-cloud-computing (visited on 04/18/2023).

[7] "Types of cloud computing". (Jul. 15, 2022), [Online]. Available: https://www.redhat.com/en/topics/cloud-computing/public-cloud-vs-private-cloud-and-hybrid-cloud (visited on 04/19/2023).

[8] L. Aleite. "Organize your azure resources effectively - cloud adoption framework". (Jan. 31, 2023), [Online]. Available: https://learn.microsoft.com/en-us/azure/cloud-adoption-framework/ready/azure-setup-guide/organize-resources (visited on 05/14/2023).

[9] bwren. "Azure monitor cost and usage - azure monitor". (Mar. 10, 2023), [Online]. Available: https://learn.microsoft.com/en-us/azure/azure-monitor/usage-estimated-costs (visited on 04/19/2023).

[10] bandersmsft. "What are azure reservations? - microsoft cost management". (Apr. 17, 2023), [Online]. Available: https://learn.microsoft.com/en-us/azure/cost-management-billing/reservations/save-compute-costs-reservations (visited on 04/19/2023).

[11] "Pricing overview—how azure pricing works | microsoft azure". (), [Online]. Available: https://azure.microsoft.com/en-us/pricing/ (visited on 05/01/2023).

[12] B. Thormundsson. "Topic: Artificial intelligence (AI) worldwide", Statista. (Apr. 27, 2023), [Online]. Available: https://www.statista.com/topics/3104/artificial-intelligence-ai-worldwide/ (visited on 05/15/2023).

[13] "Framing: Key ML terminology | machine learning", Google for Developers. (), [Online]. Available: https://developers.google.com/machine-learning/crash-course/framing/ml-terminology (visited on 05/21/2023).

[14] X. Ying, "An overview of overfitting and its solutions", *Journal of Physics: Conference Series*, vol. 1168, no. 2, p. 022 022, Feb. 2019, Publisher: IOP Publishing, ISSN: 1742-6596. DOI: 10.1088/1742-6596/1168/2/022022. [Online]. Available: https://dx.doi.org/10.1088/1742-6596/1168/2/022022 (visited on 05/21/2023).

[15] D. Berrar, "Cross-validation", in Jan. 1, 2018, ISBN: 978-0-12-809633-8. DOI: 10.1016/B978-0-12-809633-8.20349-X.

[16] C. M. Bishop, *Pattern Recognition and Machine Learning*, 1st. Springer, 2006.

[17] M. Douglas C., P. Elizabeth A., and V. G. Geoffrey, *Introduction to Linear Regression Analysis*, 5th. John Wiley & Sons, 2012, ISBN: 978-0-470-54281-1.

[18] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*, Second Edition. 2009. [Online]. Available: https://web.stanford.edu/~hastie/ElemStatLearn/.

[19] J. Brownlee. "A gentle introduction to ensemble learning algorithms", MachineLearningMastery.com. (Apr. 18, 2021), [Online]. Available: https://machinelearningmastery.com/tour-of-ensemble-learning-algorithms/ (visited on 05/01/2023).

[20] T. G. Dietterich, "Ensemble methods in machine learning", in *Multiple Classifier Systems*, ser. Lecture Notes in Computer Science, Berlin, Heidelberg: Springer, 2000, pp. 1–15, ISBN: 978-3-540-45014-6. DOI: 10.1007/3-540-45014-9_1.

[21] L. Hardesty. "Explained: Neural networks", MIT News | Massachusetts Institute of Technology. (Apr. 14, 2017), [Online]. Available: https://news.mit.edu/2017/explained-neural-networks-deep-learning-0414 (visited on 05/02/2023).

[22] H. Ramchoun, M. Amine, J. Idrissi, Y. Ghanou, and M. Ettaouil, "Multilayer perceptron: Architecture optimization and training", *International Journal of Interactive Multimedia and Artificial Intelligence*, vol. 4, no. 1, p. 26, 2016, ISSN: 1989-1660. DOI: 10.9781/ijimai.2016.415. [Online]. Available: http://www.ijimai.org/journal/node/907 (visited on 05/02/2023).

[23] M. Nielsen, *Neural Networks and Deep Learning*. 2015. [Online]. Available: http://neuralnetworksanddeeplearning.com/.

[24] D. W. Aha, D. Kibler, and M. K. Albert, "Instance-based learning algorithms", *Machine Learning*, vol. 6, no. 1, pp. 37–66, Jan. 1, 1991, ISSN: 1573-0565. DOI: 10.1007/BF00153759. [Online]. Available: https://doi.org/10.1007/BF00153759 (visited on 05/21/2023).

[25] "What is the k-nearest neighbors algorithm? | IBM". (), [Online]. Available: https://www.ibm.com/topics/knn (visited on 05/21/2023).

[26] J. Gama, I. Žliobaitė, A. Bifet, M. Pechenizkiy, and A. Bouchachia, "A survey on concept drift adaptation", *ACM Computing Surveys*, vol. 46, no. 4, 44:1–44:37, Mar. 1, 2014, ISSN: 0360-0300. DOI: 10.1145/2523813. [Online]. Available: https://doi.org/10.1145/2523813 (visited on 05/02/2023).

[27] "Skmultiflow.drift_detection.ADWIN — scikit-multiflow 0.5.3 documentation". (), [Online]. Available: https://scikit-multiflow.readthedocs.io/en/stable/api/generated/skmultiflow.drift_detection.ADWIN.html (visited on 05/11/2023).

[28] L. A. Birnbaum, *Machine Learning Proceedings 1993: Proceedings of the Tenth International Conference on Machine Learning, University of Massachusetts, Amherst, June 27-29, 1993*. Morgan Kaufmann, May 23, 2014, 361 pp., Google-Books-ID: TrqjBQAAQBAJ, ISBN: 978-1-4832-9862-7.

[29] B. Settles, "Active learning literature survey", University of Wisconsin-Madison Department of Computer Sciences, Technical Report, 2009, Accepted: 2012-03-15T17:23:56Z. [Online]. Available: https://minds.wisconsin.edu/handle/1793/60660 (visited on 05/02/2023).

[30] B. Kitchenham, O. Pearl Brereton, D. Budgen, M. Turner, J. Bailey, and S. Linkman, "Systematic literature reviews in software engineering – a systematic literature review", *Information and Software Technology*, Special Section - Most Cited Articles in 2002 and Regular Research Papers, vol. 51, no. 1, pp. 7–15, Jan. 1, 2009, ISSN: 0950-5849. DOI: 10.1016/j.infsof.2008.09.009. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0950584908001390 (visited on 05/19/2023).

[31] S. Alkharif, K. Lee, and H. Kim, "Time-series analysis for price prediction of opportunistic cloud computing resources", in Jan. 1, 2018, pp. 221–229, ISBN: 978-981-10-6519-4. DOI: 10.1007/978-981-10-6520-0_23.

[32] A. Bajaj. "ARIMA & SARIMA: Real-world time series forecasting", neptune.ai. (Jul. 21, 2022), [Online]. Available: https://neptune.ai/blog/arima-sarima-real-world-time-series-forecasting-guide (visited on 05/03/2023).

[33] R. Nabi, S. Saeed, and H. Harron, "A novel approach for stock price prediction using gradient boosting machine with feature engineering (GBM-wFE)", *Kurdistan Journal of Applied Research*, vol. 5, pp. 27–48, Apr. 30, 2020. DOI: 10.24017/science.2020.1.3.

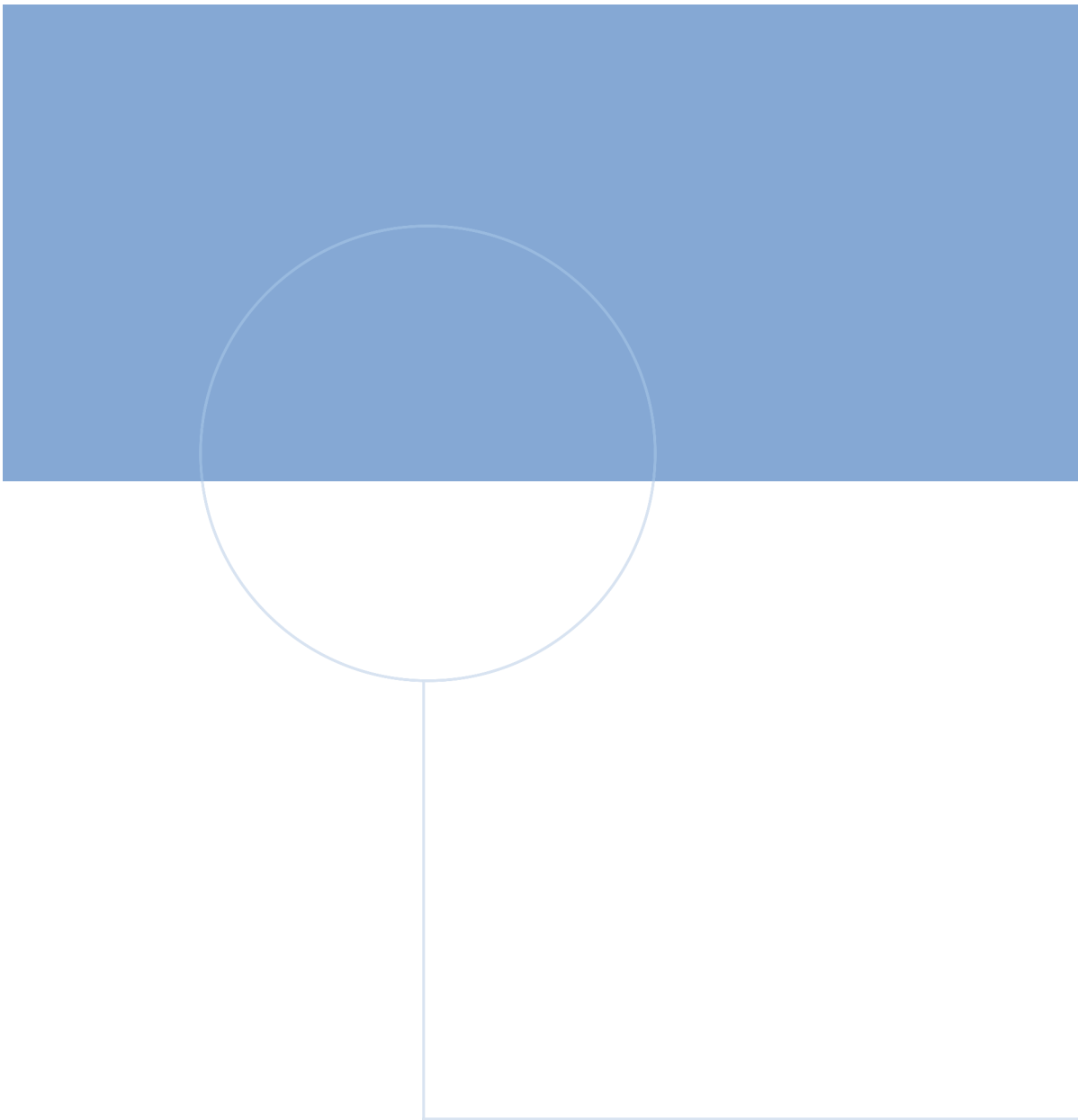
[34] C. R. Kothari, *Research Methodology: Methods and Techniques*, Second. New Age International, 2004.

[35] A. Hevner, A. R, S. March, *et al.*, "Design science in information systems research", *Management Information Systems Quarterly*, vol. 28, p. 75, Mar. 1, 2004.

[36] W. Koehrsen. "Hyperparameter tuning the random forest in python", Medium. (Jan. 10, 2018), [Online]. Available: https://towardsdatascience.com/hyperparameter-tuning-the-random-forest-in-python-using-scikit-learn-28d2aa77dd74 (visited on 05/06/2023).

8 List of Attachments

1. Project handbook
    (a) Contract
    (b) Progress plan
    (c) Meetings
    (d) Week Reports
2. Source code



NTNU

Norwegian University of
Science and Technology