YAŞAR UNIVERSITY

GRADUATE SCHOOL

NON-THESIS MASTER TERM PROJECT

TITLE OF THE TERM PROJECT

SECOND LINE OF THE TITLE

THIRD LINE OF THE TITLE

student NAME

STUDENT ID

TERM PROJECT ADVISOR: AssISt./ASSOC./PROF.(PHD) XXX YYY

DEPARTMENT

NAME OF THE PROGRAMME

APPROVAL PAGE

We confirm the suitability of the current study as a Non-Thesis Master's Term Project in terms of scope and quality.

|  |  |
| --- | --- |
| Advisor: | Signature: |
| Assist./Assoc./Prof.(PhD) Xxx YYY  … University | ....................... |
| Head of the Department: |  |
| Assist./Assoc./Prof.(PhD) Xxx YYY  … University | ....................... |
|  |  |
|  |  |
|  |  |
|  |  |

ABSTRACT

TITLE OF THE TERM PROJECT

Surname, First Name  
Non-Thesis Term Project, Name of the Programme  
Advisor: Assist./Assoc./Prof.(PhD)Xxx YYY  
Month 20XX

The maritime transportation of containers between terminals is important from the perspective of the supply chain processes. Container transportation, referred to as liner shipping, can be considered one of the most efficient and cost-effective way to carry loads for overseas transportation. Containers are handled with many operations during transportation. In the meantime, some unusual situations may occur such as the container being dropped during unloading while being handled by crane. These situations may cause the containers to become damaged or scrapped.

This paper aims to implement machine learning (ML) approaches for estimating the container damage situation. Microsoft Azure Machine Learning Studio environment has been used for the prediction model. The study results indicate that the application of two-class classification-based ML algorithms is advantageous for providing an accurate estimation of the damaged condition of containers. Therefore, constructed damage estimation model’s result is going to serve as a reference for transportation companies, aiding them in determining whether to proceed with the given transportation request and in anticipating potential damage risks.

**Keywords:** machine learning, big data, data mining, two-class classification, container damage estimation, Azure Machine Learning Studio

ÖZ

PROJE BAŞLIĞI

Soyad, İsim

Yüksek Lisans Dönem Projesi, Program Adı

Danışman: Dr. Öğr. Üyesi/Doç./Prof. Dr. Xxx YYY

Ay 20XX

Konteynerlerin deniz yoluyla terminaller arasında taşınması, tedarik zinciri süreçleri açısından önem taşır. Konteyner taşımacılığı olarak da adlandırılan bu yöntem, denizaşırı taşımacılık için en verimli ve maliyet etkin çözümlerden biri olarak kabul edilir. Konteynerler taşınırken, bir dizi operasyon aşamasından geçerek farklı yöntemlerle ele alınır. Bu aşamalarda bazı istisnai durumlar ortaya çıkabilir. Örneğin, konteynerin gemiden indirilirken vinç yardımıyla düşürülmesi veya boşaltma sırasında zarar görmesi gibi durumlar söz konusu olabilir. Bu gibi örnekler, konteynerlerin zarar görmesine veya kullanılamaz hale gelmesine yol açabilir.

Bu çalışmanın amacı, konteynerlerin hasar durumunu tahmin etmek için makine öğrenimi yaklaşımlarını kullanmaktır. Tahmin modeli için Microsoft Azure Machine Learning Studio platformu kullanılmıştır. Elde edilen sonuçlar, iki sınıflı sınıflandırma temelli makine öğrenimi algoritmalarının, konteynerlerin hasar durumlarını doğru bir şekilde tahmin etmede avantaj sağladığını göstermektedir. Yapılan hasar tahmini çalışmasının sonuçları, nakliye şirketleri için bir rehber işlevi görecek ve taşıma taleplerinin yerine getirilip getirilmemesine karar verme ve olası hasar risklerini önceden tahmin etme konusunda yardımcı olacaktır.

**Anahtar Kelimeler****:** makine öğrenimi, büyük veri, veri işleme, iki sınıflı sınıflandırma, konteyner hasar tahmini, Azure Machine Learning Studio

ACKNOWLEDGEMENTS

*(This part must be added at the project submission stage after the term project has been successfully completed.)*

Name Surname  
İzmir, 20XX

TEXT OF OATH

I declare and honestly confirm that my study, titled “TERM PROJECT TITLE” and presented as a Non-thesis Master Term Project, has been written without applying to any assistance inconsistent with scientific ethics and traditions. I declare, to the best of my knowledge and belief, that all content and ideas drawn directly or indirectly from external sources are indicated in the text and listed in the list of references.

Name Surname  
Date

**TABLE OF CONTENTS**

[APPROVAL PAGE iii](#_Toc152235096)

[ABSTRACT v](#_Toc152235097)

[ÖZ vii](#_Toc152235098)

[ACKNOWLEDGEMENTS ix](#_Toc152235099)

[TEXT OF OATH xi](#_Toc152235100)

[TABLE OF CONTENTS xiii](#_Toc152235101)

[LIST OF FIGURES xv](#_Toc152235102)

[LIST OF TABLES xvii](#_Toc152235103)

[SYMBOLS AND ABBREVIATIONS xix](#_Toc152235104)

[1. CHAPTER: INTRODUCTION 1](#_Toc152235105)

[2. CHAPTER: PROBLEM DEFINITION 3](#_Toc152235106)

[2.1. ConstructIon of the model 4](#_Toc152235109)

[3. CHAPTER: LITERATURE REVIEW 5](#_Toc152235110)

[3.1. Causes of Container Damages 5](#_Toc152235112)

[3.2. Container Damage Estimation with Machine Learning Tools 10](#_Toc152235113)

[3.3. Damage Estimation for Different Sectors Using Machine Learning 11](#_Toc152235114)

[3.4. Selecting the Appropriate Machine Learning Tool 14](#_Toc152235115)

[4. CHAPTER: DATA COLLECTION 17](#_Toc152235116)

[4.1. Data Description 17](#_Toc152235117)

[4.2. Data Analysis 22](#_Toc152235119)

[4.2.1. Data Cleansing 23](#_Toc152235120)

[4.2.2. Data Manipulation and Classification 24](#_Toc152235121)

[5. CHAPTER: METHODOLOGY 27](#_Toc152235122)

[5.1. Machine Learning Algorithms 27](#_Toc152235124)

[5.1.1. Two-Class Support Vector Machine 29](#_Toc152235125)

[5.1.2. Two-Class Average Perceptron 30](#_Toc152235126)

[5.1.3. Two-Class Decision Forest 30](#_Toc152235127)

[5.1.4. Two-Class Logistic Regression 31](#_Toc152235128)

[5.1.5. Two-Class Boosted Decision Tree 32](#_Toc152235129)

[5.1.6. Two-Class Neural Network 33](#_Toc152235130)

[5.2. Performance Metrics 34](#_Toc152235131)

[5.2.1. Accuracy 35](#_Toc152235132)

[5.2.2. Precision 36](#_Toc152235133)

[5.2.3. Recall 36](#_Toc152235134)

[5.2.4. F1 Score 36](#_Toc152235135)

[6. CHAPTER: COMPUTATIONAL EXPERIMENTS & RESULTS 39](#_Toc152235136)

[6.1. Initial Results of Experiments 39](#_Toc152235138)

[6.2. Results After Parameter Setup 43](#_Toc152235139)

[6.3. Discussion of Results 47](#_Toc152235140)

[7. CHAPTER: CONCLUSIONS AND FUTURE WORK 51](#_Toc152235141)

[8. REFERENCES 55](#_Toc152235142)

[APPENDICES 59](#_Toc152235143)

[APPENDIX 1. RESULTS OF THE EXPERIMENTS 59](#_Toc152235144)

[APPENDIX 2. SQL QUERY FOR GATHERING BIG DATA 65](#_Toc152235145)

LIST OF FIGURES

[**Figure 1.1.** Example of Voyage of the Vessel and Terminal Calls 1](#_Toc145498823)

[**Figure 4.1.** Harmonized Code System: Coffee Example. 20](#_Toc145498824)

[**Figure 4.2**. Damaged = YES/NO Classification on Constructed SQL Query. 22](#_Toc145498825)

[**Figure 5.1.** Working Principle of Two- Class SVM Algorithm. 29](#_Toc145498826)

[**Figure 5.2.** Working Principle of Two-Class Average Perceptron Algorithm. 30](#_Toc145498827)

[**Figure 5.3.** Working Principle of Two-Class Decision Forest Algorithm. 31](#_Toc145498828)

[**Figure 5.4.** Working Principle of Two- Class Logistic Regression Algorithm. 31](#_Toc145498829)

[**Figure 5.5.** Working Principle of Decision Tree Algorithm. 32](#_Toc145498830)

[**Figure 5.6.** Working Principle of Artificial Neural Network Algorithm. 33](#_Toc145498831)

[**Figure 6.1.** Tune Model Hyperparameter Module in Azure ML (Classic) 43](#_Toc145498832)

[**Figure 6.2.** Constructed Final Azure ML Experiment with 6 Algorithms. 44](#_Toc145498833)

[**Figure 6.3.** Split Data Module in Azure ML (Classic). 45](#_Toc145498834)

[**Figure 6.4.** Azure ML Webservice Test Interface for Entering Feature Inputs. 49](#_Toc145498835)

[**Figure 6.5.** Azure ML Webservice Test Result Output. 49](#_Toc145498836)

LIST OF TABLES

[**Table 4.1.** Two Example Observations in the Dataset 22](#_Toc145498837)

[**Table 5.1.** Confusion Matrix 35](#_Toc145498838)

[**Table 6.1.** First Iteration Results of Six Algorithms 40](#_Toc145498839)

[**Table 6.2.** Second Iteration Results of Six Algorithms 40](#_Toc145498840)

[**Table 6.3.** Third Iteration Results of Six Algorithms 41](#_Toc145498841)

[**Table 6.4.** Fourth Iteration Results of Six Algorithms 41](#_Toc145498842)

[**Table 6.5.** Fifth Iteration Results of Six Algorithms 42](#_Toc145498843)

[**Table 6.6**. Sixth Iteration Results of Six Algorithms 44](#_Toc145498844)

[**Table 6.7.** Split Data Module Experiment Alternatives via Static Design of Experiment 46](#_Toc145498845)

[**Table 6.8.** Best Experiment Result Between a Total of 17 Experiments 47](#_Toc145498846)

[**Table 6.9**. Summary Table of Main Iterations and Results 48](#_Toc145498847)

SYMBOLS AND ABBREVIATIONS

SYMBOLS:

TN True Negatives

TP True Positives

FN False Negatives

FP False Positives

F1 F1 Score

ABBREVIATIONS:

ERP Enterprise Resource Planning

API Application Programming Interface

ML Machine Learning

SVM Support Vector Machine

SVR Support Vector Regression

SVC Support Vector Classification

NN Neural Network

CNN Convolutional Neural Network

DB Database

NN Structured query language

POR Place of Receipt

POL Place of Loading

POD Place of Discharge

DEL Delivery Point

T/S Transshipment

HS Harmonized Code Notation

# CHAPTER: INTRODUCTION

Maritime transportation has been one of the viable solutions to the global transport of goods. Around 80% of international trade is carried out by naval carriers handled by the ports and terminals (Unctad, 2020). Liner shipping is one of the most efficient and cost-effective maritime transportation nodes, representing a substantial part of the global economy.

The world is becoming more global and the availability and accessibility of goods in the global economy are becoming more critical. In today's competitive environments, many container vessels can carry many containers of goods only in a single “Voyage”. In detail, the liner shipping operations can be compared with the cities’ public bus transportation systems. In the liner shipping transportation process, there are “Voyages” that the Vessels are assigned on. As shown in Figure 1.1, vessels call terminals with the same frequency and order and these routings are called “Voyages”.

Map

Description automatically generated

**Figure 1.1.** Example of Voyage of the Vessel and Terminal Calls

Source: Author

These Voyage Services of Liner Shipping Companies realize significant container transport volumes daily. Therefore, these containers experience some standard operations or encounter some unusual situations. The containers are faced with many damage or scrap risks. Especially in less developed countries the risk factor is increasing.

The container damage estimation tool is developed using the machine learning approach in this scope. Data cleaning, manipulation, and classification processes are performed in the first stage. Then, iterative runs with alternative algorithms are performed and these results are evaluated based on the determined performance criteria.

The rest of this study is organized as follows: Chapter 2 presents the problem definition, Chapter 3 gives the literature review under four main topics, and Chapter 4 explains the construction of big data and the data mining processes. Chapter 5 introduces the machine learning algorithms and their performance metrics for measuring the success of the algorithms. Chapter 6 includes computational experiments in different phases and a discussion of the results. Chapter 7 gives the conclusion and direction for future work on this subject.

# CHAPTER: PROBLEM DEFINITION

Container shipping companies, in other words, liner shipping companies transport vast amounts of containers between countries. Within these transportation operations, these liner shipping companies get many services such as transshipment operations, stevedoring operations, loading & discharge operations, container lock & unlock operations, and intermodal truck & rail transportation operations.

All the different transportation mode changes and services include many risks for container conditions and security purposes since containers can be damaged or scraped by humans and other factors in the supply chain. Each step of container transportation includes different risk factors, for instance, transshipment operations of the containers include stevedoring operations, and each stevedoring function has a transportation process with cranes which can be risky for the container itself or the commodity inside of the container.

Therefore, estimating the container damage and scrap risk for the container's transportation process in advance will benefit the liner shipping companies and container owner (lessor) companies. After finding out the most significant damage and scrap risks, the input causes for the damage condition of the containers are generated and an estimation model is constructed with different machine learning algorithms. This way, customers' transportation needs are met with proper container usage and minimized repair costs for the container owner company. For instance, if the damage estimations of the containers are accurate, then the container owner company can decide the most appropriate container in its stocks for that transportation in terms of the container condition.

In summary, container shipping companies and lessor companies can protect their containers from damage and scrap risks by accurately estimating their damaged condition.

For the estimation of the damage condition and proper selection of the containers, the construction of the machine learning model is completed and as a future work, the developed model will be integrated into an Enterprise Resource Planning (ERP) application for daily operational purposes.



## ConstructIon of the model

The container damage estimation model is developed by experimenting with different machine learning algorithms.

* Data collection & preparation steps are followed.
* Machine learning methods and algorithms are used for the determination of the damaged condition of the container below:
  + Two–class classification machine learning algorithms are used for answering simple two-choice questions and making forecasts by estimating the “Yes or No”, or “True or False” conditions like the state of the container “Damaged or Non-damaged”.

# CHAPTER: LITERATURE REVIEW

The literature review is organized into four categories. The first research topic focuses on identifying the causes of container damage during transportation. The second topic examines studies related to container damage estimation using machine learning models. The third category reviews prior research on damage condition estimation across various sectors using machine learning tools. Lastly, the review delves into the selection of appropriate machine learning models and methods to prevent overfitting. Since the correct model selection is important and overfitting is one of the main problems that can be encountered while working in the machine learning environment.



## Causes of Container Damages

In the first step of the literature review, the main risk factors affecting the containers' damage situation have been investigated from different perspectives.

The first study was published by Chang, C. H et al. (2015), The primary purpose of this paper is to explore and analyze the risks in container shipping operations from a logistics perspective. The paper identifies and analyses risks associated with the three flows in logistics − information, physical, and payment flow. According to the previous literature research, the author found that, under the physical flow risk category, the risk elements and factors which are related to the container damage are determined as;

* Risk elements are cargo/asset loss or damages.
* Risk factors are Damage to containers or cargo due to terminal operators’ improper loading/ unloading operations, Cargo being stolen from unsealed containers, Damage caused by transporting dangerous goods, Damage to ship or quay due to improper berth operations, and Attack from pirates or terrorists.

Risk factors greater than 10 points can be considered in the high-risk category. Among the 35 risk factors, the elements have been ranked according to their risk factor points. When the determined container damage-related points are examined, it is found that all points are in the top 10 risk list.

Cyclone risk model and assessment for East Asian container ports has been shared by Wei, J. et al. (2019), and the study mainly aims to provide a more comprehensive vulnerability assessment of critical port infrastructure to cyclone impact so that the potential damage and loss can be quantitatively evaluated.

In this study, cyclone event risk factors have been examined for the 14 different East Asian ports. Cyclone events are typically associated with strong winds, heavy rainfalls, wave actions, storm surges, and coastal flooding. But within this study’s scope, the wind and cyclone-induced storm surge, and cyclone events have been analyzed more detail. Physical vulnerability is the expected damage or loss percentage of asset values under the given hazard intensity to evaluate the quantitative risks. Under the storage facilities branch the container damage state/loss ratio also has been considered in the vulnerability assessment.

According to the vulnerability assessment, cyclone events can also cause container damage. Therefore, the ports' weather and climate conditions can be considered input for the container damage estimation model.

Tseng, W. et al. (2013) have published this study to determine the cargo damage risk factors and study the cargo damage risk management in export operations. The risk factors for ocean freight forwarders have been selected via expertise in the area and a literature review. According to the determined risk factors, the below points seem to be related to the damage to the container:

* Delay or damage caused by throwing cargo
* Poor transport packaging
* Land traffic accidents
* Cargo damage during port loading/unloading or the marine transport
* Poor arrangement of cargo in container

The risk prevention or reduction recommendations have been shared for all determined risk factors. Then, the risk factor and the recommendations were combined in a matrix to show the recommendations and the risk factor relations. The all-container damage-related risk factors seem to have a medium risk level on frequency- severity risk matrix. When the cost-benefit matrix of the related points is considered, their cost-benefit ratios seem average. The implementations can be costly, but their return on investment in risk reduction or prevention cannot be negligible.

From the risk factor findings of the paper, it can be quickly concluded that the packaging of the commodities, loading / unloading activities, cargo planning of the container, and inland point transfer of the container for the evaluation of the traffic accident condition can affect the damage state of the containers.

Shang K.C. et al. (2010) introduced a study of container terminals in Kaohsiung Harbor. They determined risk factors in all stevedoring operations, such as loading and unloading operations of container terminals. The risk analysis has been implemented for three different terminals. The frequency of the risk has been categorized between 1 to 5, 1 shows very rare events, and 5 shows too regular events. On the other hand, the severity of the risk is measured with the costs. They are scaled from 1 to 5. 1 is the least important, and five is the most important.

Risk factors that are related to the container damage are determined as follows:

* Inappropriate use of the 40- and 20-foot hangers and the unbalanced overweight containers which caused the moving containers to collide.
* The switch of hatch covers did not cooperate well with conductors and the 40- and 20-foot hangers were inappropriately used.
* Because of damage to the hanger of the gantry crane, the container locks fell into the trailers.
* Negligence of duty operators (Fast raising or inadequate gantry crane height).
* The unreturned suspenders on the ships and the gantry crane collide with the suspenders while loading and unloading.
* Because of climatic factors (typhoons), facilities do not take any precautions to prevent collisions.
* Slipping the containers to the trailers directly.

When the container damage-related risk factors have been analyzed, all damage-related points are located around the medium risk level. Therefore, the risk analysis results show container damage risks in stevedoring operations. When the determined related risk factors have been evaluated, these points can be the factors that can directly affect the container damage state: the gantry cranes and their terminal details, operator & shift details, container commodity details, hatch cover amounts, and climate factors.

A model has been created for quantitatively analyzing the container shipping operational risks, which has been shared by Nguyen, S. et al. (2019), After analyzing the quantitative risks, the Bayesian Network method determines the risk levels.

The parameter structure for the calculation of the risk level can be explained in classes as follows, Secondary parameters: Financial Impact, Reputational Impact, Operational Impact, Risk Being Undetected, Detection Lateness and the Primary parameters are Likelihood of Occurrence, Severity of Consequences, Level of Detectability. All these secondary parameters and determined three levels (low, medium, high) linguistic grades in all individual states have been defined. In the following step, the risk factors are collected from the domain experts and the container damage related has been determined as follows:

* Inland traffic accidents and system inefficiencies.
* Maritime accidents.
* Damage to containers or cargo in cargo handling operations.
* Damage caused by transporting dangerous goods.
* Damage to reefer containers due to electricity failure.
* Cargoes or vehicles being stolen or tampered with.
* Piracy and terrorism.

After data entries of all determined factors, each risk level has been calculated according to the parameters. Risk factors are ordered and ranked. And it is found that the out of seven factors can be ended up with container damage, five of which are in the top 15 risky factors rank. This concludes that the damage of the container-related points includes high-risk container shipping operational risks. From these findings, the inland point transfers, maritime accidents, stuffing-unstuffing activities of the containers, dangerous goods commodity transportation, reefer electricity failure, piracy & terrorism can be effective inputs for the container damage estimation model.

Ellis, J (2011) has aimed to determine the dangerous goods accident factors, and dangerous goods accidents to overall container ship accident rates. When the collected data has been analyzed, the number of dangerous goods transportation accidents in the UK is around 1-3 per year. But in the US, it is observed that the accident number can go up to 12-13 per year. When the fatality level of the accidents is considered, these numbers also show the importance of dangerous goods transportation. The reason for the accidents which cause the damage to the containers are:

* Cargo characteristics outside the appropriate range: impurities, not cooled sufficiently, etc.
* Container/Packaging deficiencies: valve or vent problems, pre-existing damage, unclosed valves, overfilling.
* Poor securing, bracing, blocking, etc. inside the container leads to damage.
* Loading/unloading faults. (e.g., drops, contact)
* Inappropriate stowage of goods in terms of temperature, ventilation, and ignition source.
* Improper stacking loads and procedures.

When the reasons are evaluated, the dangerous good accident causes indicate that the cargo characteristics, overfilling, poor securing, loading/unloading operations, stowage and stackings can be essential inputs for the container damage situations.

Singh S. P. et al. (2014) discussed improperly loaded trailers and containers. They recommended the most proper loading methods to minimize the damage and injuries related to the loading and packaging operations. For a more secure load of the containers and trailers, the standard methodologies given below should be adopted:

* Large and heavy freight should be placed on the bottom.
* If the trailer is partially loaded, a staircase method may be used to step down the freight to the rear of the trailer.
* Trailers should be loaded tightly.
* Void spaces should be avoided in lateral and longitudinal directions inside closed van trailers, intermodal containers, and boxcars.
* Additional devices such as wooden blocks, load bars, other freight, dunnage, airbags, or friction rubber mats may be used based on industry standards that provide safe loading methods.

Within all these findings, the loads' packaging and weight distribution are essential for the container damages since they directly affect the load securement and the damaging conditions for the container.

Wan, C et al. (2019) introduced a model which aimed to access the risk factors with the capability of improving result accuracy under a high uncertainty in risk data.

According to the main risk factors that the shipping company determines, the most critical risk factor seems that the cargo of dangerous goods loads. From there, it can be quickly approved that hazardous goods transportation should considered for the container damage estimation models as an input of the model.

Container damage reasons are investigated in this literature review step. Chang, C. H et al. (2015), Wei, J et al. (2019), Tseng, W. et al. (2013), Shang K.C. et al. (2010), Ellis, J (2011), Singh S. P. et al. (2014), Wan, C et al. (2019) all of the researchers have found and determined the cargo and container damage risks and reasons in their study such as, the negligence of duty operators, the hanger of the gantry crane, the container locks falling into the trailers, improper loading/ unloading operations, Damage caused by transporting dangerous goods, Cargo characteristics outside the appropriate range. According to the determined cargo and container damage risks, all findings are evaluated for the feature selection of the container damage ML model as a “Reason” for the damage by combining professional comments of the business responsible.

## Container Damage Estimation with Machine Learning Tools

This part of the literature review presents the review of the Container damage estimation with machine and deep learning usage in the literature.

Ekici, I. Z. et al. (2020) closed the literature gap by automating the damage detection of containers via a convolutional neural network (CNN) method with the support of image recognition technology.

The damage situation of the container has been tried to be estimated via the CNN method. CNN method is an artificial neural network model which is highly preferable for image recognition actions. From there, it can be quickly concluded that the damage situation of the container can be estimated via ML methods even if the image recognition process has been included.

When the scope of the paper and our study's scope have been benchmarked, they seem to be similar. But this study aims to detect the damage situation of the containers after the occurrence of damage situation by using image recognition technology. This study aims to predict the damage situation of the container before it occurs by using two-class classification machine learning methods.

Additionally, it is explained that "Deep Learning" is an optimization problem. Different methods are used for finding the optimum value for nonlinear problems. This issue should be considered within the study's scope.

As an output, the damaged situation of the container has been determined with two classes or binary approaches, "Damaged / Undamaged". The damaged state indicates 1, undamaged state indicates 0 according to binary notification. Also, this study’s scope parallel approach will be constructed, and the predictable output will be converted as binary notification.

Container Damage Estimation studies were scrutinized under this step of the literature search study. Ekici, I. Z. et al. (2020) have applied the CNN method, an artificial neural network model widely used for image recognition actions. Within this scope, aims to detect the damaged containers via the image recognition-based CNN method. This study indicates that ML studies can be applied to the damage situations of the containers. The main difference between this study and Ekici, I. Z. et al. (2020) study is that the damage occurrences are tried to be detected before it happens by using two-class classification-based ML models.

## Damage Estimation for Different Sectors Using Machine Learning

In this part of the literature review, estimations of the damage situations via machine learning approaches in other sectors have been investigated. Additionally, as mentioned in the objectives section, the machine learning techniques are used with “Two-class classification” machine learning algorithms. Accordingly, the other sector examples of this algorithm, such as Support Vector Machine (SVM), Anomaly detection, and Classification models, have been examined and summarized.

Hart, E. et al. (2019) has studied the creation of machine learning models to model wind damage to forests. The paper's main objective is to estimate the storm-related damage situations of forest trees. Different machine learning models are constructed to estimate the "Damage" or "No Damage" situation within the scope. Similarly, container damage situations can also be evaluated within the same states: "Damage" or "Not damage".

Random forests (RF) are a more recent technique when compared with other methods, and their success in developing models from noisy and unbalanced data is proven. The RF algorithm constructs independent decision trees whose results are combined to make an estimation. The technique has an advantage in speed to train. Therefore, they are becoming extremely popular in many aspects of forest modelling. Logistic regression models have been used to assess the risk of wind damage since their dependent variables are categorical. If the dependent variable (output) is binary like 0 or 1, they are ideal for damage prediction like damaged or undamaged. Additionally, logistic regression models can be used to identify which features are associated with the damage.

In summary, the random forest and logistic models can be used for damage estimation since both successfully determine the binary outputs such as damaged / undamaged. Cross-validation and data-balancing methods can be used for the data preparation step because the number of damaged and not-damaged observations is not close. From all these findings, for the container damage estimation, the mentioned models can be taken into consideration.

Wei, J. et al. (2020) show the application of the Lease-squares Support Vector Machine (LS-SVM) methods with SCOTE data sampling methodology in bearing fault diagnosis in their study.

SVM is based on structural risk minimization & principle of VC dimension theory in statistical learning. SVM has advantages in small samples, non-linear, and high-dimensional pattern recognition problems such as prediction of the damage situation of the containers.

On the other hand, SVM & LS-SVM methods can fail when the data is imbalanced. Therefore, data balancing methods should be applied in the data preparation step, or SVM methods should be used more carefully.

Feng, Y. et Al. (2020) introduced Stochastic nonlocal damage analysis by a machine learning approach: the "Capped-extended-support vector regression" method (C-X-SVR) in their research.

Within the damage analysis scope, two main models have been examined. These are deterministic damage models & stochastic nonlocal damage analysis of quasi-brittle material. Then, the machine learning technique (C-X-SVR) is included in real-life stochastic damage analysis. This shows that ML techniques can be applied in the damage analysis process.

The C-X-SVR model extends the traditional Support Vector Machine technique which extends the binary classification to regression problems. The main advantage of the model is that it can guarantee computational durability. With mentioned scope, the proposed model can be evaluated within the scope.

A total of 3 experiments & applications have been used to validate the proposed method. Between the results, the mentioned Support Vector Machine approach seems to have superior capability towards estimating the experimental dataset.

Patil, K. et al. (2017) aimed to develop a car damage classification method with deep learning techniques since the claim leakage issue is essential for the car industry.

The study started with the classification of car damage. Seven main damage classes have been constructed and the classification experiment has been conducted with an 80-20 training-test ratio.

From there, it can be summarized that the classification approach is considered according to the variety of a feature or an output. Since this study is evaluated under the "Classification" experiment, the 80-20 train-test ratio can be followed if the classification method has been applied.

Kulkarni, H. et al. (2020) provided storm-level forecasting by using the support vector machine method within this paper’s scope.

While determining the dataset some features are classified and can take low, medium, serious, and high classes to increase the model's prediction strength. The data pre-processing methods are used in the data preparation step such as cleaning, transforming, and sampling of the data. Since the principal purpose of the preprocessing is enhancing the quality of the data, this step seems a "must" before proceeding through the algorithm running step.

Support Vector Machine (SVM) is becoming popular in machine learning. SVM performs classification by creating an n-dimensional hyperplane that divides the data into two groups. The support vector clustering (SVC) function performs multi-class classification on a dataset.

This literature has focused on various sectors' damaged condition estimation studies via ML. Within these sectors, some of the studies use parallel two-class classification ML methods for providing an estimation. Hart, E. et al. (2019), Wei, J. et al. (2020), Patil, K. et al. (2017) applied two-class classification ML methods to predict damage and fault situations about the determined issues. This proves the two-class classification methods are appropriate for estimating the containers' damaged state. As an additional finding, Feng, Y. et al. (2020), and Kulkarni, H. et al. (2020) used multi-class classification and other methods to detect the mentioned subjects. Additionally, the used algorithm approaches are interpreted in this study’s ML algorithm selection decision.

## Selecting the Appropriate Machine Learning Tool

The last step of the literature review includes essential points while selecting the appropriate machine learning models and due to the potential complexity of the data, preventing the overfitting problem while training the problem.

In this paper, a brief review of various ML algorithms has been provided by Ray S. (2020). The mentioned algorithms include only the popular ones. Mainly, the algorithms and their application purposes are mentioned for deciding the selection of an appropriate learning algorithm to meet the specific requirements of the area of application. According to the given application purposes and area of applications in the paper, the state of the container can be evaluated under the classification problem. Because the state is held as "Damaged / Undamaged". The logistic regression method can be used for the classification problem. The other method is the decision tree; the decision tree also can be used in classification problems with a special type of tree: "Classification Tree". The SVM method can handle both regression and classification problems. Since the SVM method is used under two-class classification algorithm sets in this study’s scope.

Caruana, R. et al. (2000) has pointed out the overfitting issue in neural networks. The networks with excess capacity are generalized as successful when trained with backpropagation and early stopping. The conducted experiments mainly suggest two main reasons for this:

* Overfitting can change significantly in different fields of the model. Excess capacity provides a better fit to regions with non-linearity and backpropagation can provide avoidance of overfitting the regions with non-linearity.
* Independent of the size, nets are learning subcomponents in similar sequences.

Overfitting can occur when the model's training has been started and the difference between validation loss and test loss increases followingly. When the validation error decreases, the training has been stopped and the previous state has returned. So, the large networks can be trained with early stop and back-propagation without providing significant overfitting. The primary purpose of early stopping is to stop at the ideal point, not at a specific ending point. An overfitting issue can occur since the study has included a potential network complexity. The main disadvantage of overfitting is that it provides good performance on the training data but poor generalization for other data.

Srivastava, N. et al. (2014) found a technique to prevent neural network over-fitting issue: Dropout. The main logic behind the technique is randomly dropping units from the neural network during training. With this logic, the co-adoption between units has been prevented. The method significantly reduces overfitting and provides significant improvements over other regularization methods.

The dropout method can be implemented for fully connected layers. The model’s success can be increased by disabling the nodes under the threshold value. While training with the dropout technique, neurons become more independent and efficient.

The last literature review is about the suitable algorithm selection and important details. Ray S. (2020) clarified the ideal ML algorithms for the two-class classification type datasets. Ideal algorithms are offered with advantages and disadvantages. These are Logistic regression, SVM, and Decision tree methods. On the other hand, Caruana, R. et al. (2000), and Srivastava, N. et al. (2014) have pointed out the “Overfitting” risks of the ML algorithms and prevention methods from “the Overfitting” situation. Under these findings, the optimum algorithm set for the two-class container damage prediction study and their risks is adopted with these studies.

# CHAPTER: DATA COLLECTION

From actual observed data, all previous observations have been generated from an Enterprise Resource Planning (ERP) application of a Liner shipping company. The collected data has been studied in two steps: Data Description and Data Analysis.

In the data description step, all potential and collectable features of the models have been determined and all data has been collected. After generating the dataset with determined features, data has been finalized for the training runs of machine learning algorithms.

In the data analysis step, the collected data has been merged and features have been classified for converting the elements as an operable input for the ML models. Data cleansing methods have been applied to ensure the dataset is clean and includes no missing or wrong data. Subsequently, because of the large size of the dataset outlier observations have been deleted according to the specified feature observation limits.

After completing the data analysis step, a total of 3.040.909 observations have been prepared for the two-class classification ML model studies.

## Data Description



The big data has been prepared from the database (DB) management software, Oracle TOAD SQL environment (Quest, 2023). One of the biggest maritime liner container transportation companies in the Mediterranean market provides the data source. As mentioned, data has been stored in DB and by using SQL query techniques, the related features are gathered from the ERP software of the said company’s DB.

Constructed SQL data file includes more than three million observations between 2016-2020. In other words, the 4-year range has been determined as an input for the model. According to the business expert, the main reason behind selecting these years as an input for the ML model is that these years' observations are healthy, uncorrupted, and have few missing data. Additionally, according to the interviews with the business responsible, the selected year range interprets container transportation volume trends, monthly, seasonal, and extreme conditions.

Every shipping company which owns or leases containers holds container movement history in their system with an application that shows the current/past locations of the container, realized movements, and the status of the container such as empty – damaged – fully loaded. Additionally, inland transportation (before/after vessel load-discharge of the container) included/excluded situation, transshipment condition of the container, and used vessel–service information for that transportation. The primary big data is collected from mentioned equipment history source to give much more accurate input for the machine learning model about the damage or non-damage conditions of the container.

Features are independent variables given as input for the machine learning models. In this study, as a result of the discussions held with several business departments of the liner container transportation company, the independent variables (features) were determined. These features can affect the damaged condition of the container. The determined features are:

* **Sail – Year:** This feature shows the year of the transportation. This variable has an importance for the damaged condition of the container. Because, the year information correlated with yearly vessel port call frequency, yearly container transportation trends, amounts, and volumes.
* **Sail – Month:** The month features exhibits, the month information that the container was carried. The importance of this feature is parallel with the sail-year feature. Additionally, this feature holds more details such as seasonal – monthly transportation demand trends and amounts and seasonal weather conditions.
* **Port of Loading (POL)-Country:** The port of loading country indicates the container’s loading port country. Loading operations in ports include many risky physical movements that can cause damage to the container. For example, crane usage while loading containers on the vessel, repositioning containers in ports from one depot to another via different vehicles, lashing – unlashing operations, cargo loading – discharge operations in the container at customer depot or port depot field. With the help of this feature, it is believed that the model's accuracy will be increased.
* **Port of Discharge (POD)-Country:** The port of discharge country presents the container’s discharge port country. The same risks are valid for the discharge operations as stated in the POL-Country feature.
* **Place of Receipt (POR):** This feature is classified as a “Binary” value in the model. It can get “YES / NO” values in the collected observation and represents whether the inland point to port land transportation is included in the loading/export side of the transportation or not. As might be expected, land transportation can carry some risks for damage to the container for instance, traffic accidents while carrying containers on trucks and loading/discharge operations on trucks.
* **Delivery Point (DEL):** The delivery point indicates the same situation as the POR feature. But the only difference is that land transportation is done for the discharge/import side of the transportation for this feature. The same risks can be highlighted for this feature.
* **Number of Transshipments (T/S):** The number of transshipments represents the count of the transshipment of the container during the transportation period. This feature can get an integer value between 0-4, in which 0 shows the direct transportations without T/S. 1-4 shows the number of T/Ss for that transport sample. 4 is the maximum available T/S for one transport. The feature is crucial for the ML model because in T/S operations container experiences the same risks as in loading–discharge operations. And it is expected that when the number of T/S increases in transportation the damage risk will increase since the container's potential handling will improve.
* **Packing & Unpacking Place:** These two features show the location of the loading or unloading place of the containers. This information indicates the damage risk of the container while packing the containers before loading on the vessel or unpacking the containers after discharge from the vessel. There are two location alternatives for realizing packing-unpacking operations: Customer location (Shipper & Consignee), Terminal location’s loading field.
* **Service:** Service indicates the liner container transportation company’s pre-determined and often set as fixed port call sequences. “Service” from loading port A to discharge port B in Voyages of the Services. This feature includes a fixed rotation- port call sequence order and some services can consist of more risky locations such as less developed countries’ terminals.
* **Vessel:** Liner shipping companies must operate vessels in their services to carry the containers for transportation purposes. Liner shipping companies can own vessels or can be chartered. The vessel information is essential because vessel's age, size, and specialty are important for the container damage situation. Whether the vessel has a crane can be an example of input for the model.
* **Commodity Group:** Commodities represent the cargoes that are loaded in containers. In the liner transportation sector, the commodities have a global standard to manage cargo operations such as customs procedures with a common language. This standardization is met with Harmonized Code (HS) system (International Trade Administration, n.d.). Each commodity has a unique HS Code number, and all these HS Codes are merged into standardized HS Code groups. The coffee commodity example is given in Figure 4.1. Since the containers are produced for carrying commodities, the commodities group can be counted as crucial input for the model.

metin, ekran görüntüsü, diyagram, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

**Figure 4.1.** Harmonized Code System: Coffee Example.

Source: Cogoport, 2021

* **Total Net Weight:** Containers have determined maximum payload limits by their manufacturers according to their specialties. Loading commodities close to these payload limits can decrease the resistance of the containers or can create metal fatigue which can damage the containers. Therefore, this feature is also one of the most critical inputs for the ML model.
* **Container Size / Type:** Containers have different styles and sizes for meeting several commodity transportation demands. Container sizes can be 10 feet(ft), 20-ft, 40-ft, 45-ft long. The most used container sizes are 20 ft (6.09 meters) and 40 ft (12.19 meters). The containers also have several types such as reefer containers including refrigerator systems for food loads, general-purpose containers for all commodities, and open-top containers for out-of-gauge commodities. These features are directly related to the damaged state of the containers.
* **Container Age:** All containers’ have a registration certificate that the container owner shipping companies hold. The difference between the production and collected data years shows the container's age. The age of the container directly affects its durability and is related to the damage of the containers for the vast amount and count of loads in the transportation cycle.
* **The Number of Previous Damages:** Liner shipping companies follow their containers and damage situations via holding container movement histories. In the records, there are damage and repair condition codes. By counting these codes for each specific container, the number of previous damages of a single container can be accessible. It can be easily argued that the damage and repair cycles of the containers are decreasing the durability and resistance of the containers from the cargo loads and all transportation-related operations. This input seems to be critical for the success of the model.
* **The number of Previous Full Loads:** The container movement histories show information about the previously completed transportation of the container. Therefore, the number of previous full transports can be counted by using this historical data. This parameter indirectly indicates the metal fatigue and damage probability of the container since each full transportation can generate stress on the container and creates abrasion on the container which can end up with damage to the container.

## Data Analysis

In the data analysis step, initial data has been gathered from the mentioned liner company’s ERP software. The SQL query has been constructed according to the determined feature inputs and according to the features past damage situations and observations have been tagged with 0 and 1 indicating “No Damage” and “Damage” respectively. The SQL transformation of this classification is presented in Figure 4.2.

metin, ekran görüntüsü, yazı tipi, beyaz içeren bir resim

Açıklama otomatik olarak oluşturuldu

**Figure 4.2**. Damaged = YES/NO Classification on Constructed SQL Query.

Source: Author

Between 2016-2020, more than 3 million data observations have been collected. Therefore, big, and raw data for running has been prepared. Example raw data can be examined in Table 4.1.

**Table 4.1.** Two Example Observations in the Dataset

|  |  |  |
| --- | --- | --- |
| **Features** | **Example #1** | **Example #2** |
| SAIL YEAR | 2017 | 2016 |
| SAIL MONTH | MARCH | JANUARY |
| POR HAULAGE (Y/N) | NO | NO |
| DEL HAULAGE (Y/N) | YES | NO |
| POL COUNTRY CODE | RO | BG |
| POD COUNTRY CODE | ES | IT |
| NUMBER OF TRANSSHIPMENT | 1 | 1 |
| VESSEL | ADD | FDS |
| SERVICE | SAS | SED |
| EQP SIZE | 40 | 20 |
| EQP TYPE | HC | GP |
| EMPTY/FULL (E/F) | F | F |
| TOTAL NET WEIGHT (KGs) | 3618 | 26818 |
| NET WEIGHT CLASS | LIGHT | HEAVY |
| AGE | 5 | 18 |
| **Features** | **Example #1** | **Example #2** |
| AGE CLASS | 0-5 | >15 |
| COMMODITY GROUP | MISCELLANEOUS | MATERIAL |
| PREVIOUS DAMAGES | 2 | 3 |
| PREVIOUS FULL LOADS | 10 | 56 |
| PACKING PLACE | CUST | PORT |
| UNPACKING PLACE | CUST | CUST |
| **DAMAGED** | NO | YES |

The initial data analysis step shows that the raw data includes missing and outlier data examples. Therefore, initial data needs to be processed with data mining methods.

### Data Cleansing

In the data cleansing step, all features have been investigated, and following cleaning methods, missing or wrong value data rows have been cleaned.

* Records with null values in POR – DEL haulage location features have been cleaned.
* Records with null and negative Net weight values are deleted. Because, logically, the net weight value must be greater than or equal to zero.
* The observations with missing container age information have been cleaned.
* The confusion matrix has been examined after the first run of the Machine learning model and the data which include missing values in its features have been deleted by benchmarking the total counted dataset in the confusion matrix versus the actual dataset.
* Only the “Net Cargo Weight” feature has been investigated in the outlier cleaning step. The data with a Net cargo weight greater than 30.000 kgs is evaluated as an outlier since a full container can carry a maximum of approximately 25.000 kgs. Therefore, they are assessed as outliers and cleaned from the dataset.

### Data Manipulation and Classification

In the data analysis step, the dataset’s features have been manipulated by using classification methods. The following features have been classified to be able to address and label the feature values. Mainly, numerical features scattered over a wide range are classified for increasing the model's success.

* Net Cargo Weight: This feature includes many values in the observation dataset. Therefore, the following ranges are determined with the related business responsible for classifying this feature to increase the model success, and the multi-label classification method has been applied.
  + Total Net Weight = 0, Gets “EMPTY” label
  + 10000>= Total Net Weight >= 1, Gets “LIGHT” label
  + 20000>= Total Net Weight >= 10001, Gets “MEDIUM” label
  + Total Net Weight >= 20001, Gets “HEAVY” label
* Container Age: This numerical feature includes the age information of the container in terms of the year. Since this feature can get different amounts, a multi-label classification method is implemented with the help of the responsible.
  + 5>= Container Age>=0, Gets “0-5” label
  + 10>= Container Age>=6, Gets “6-10” label
  + 15>= Container Age>=11, Gets “11-15” label
  + Container Age> 15, Gets “>15” label.
* Commodity Group: Commodities show the goods information which is carried in containers. There are approximately 100 groups according to the HS Code standards. [HS Code link]. One hundred different group alternatives can generate a complication in the ML model and reduce the success of the model. Therefore, it should be classified as a model success. The multi-label classification method is applied according to general Commodity Group classification standards. (Foreign Trade, n.d.)
  + Commodity Group between 01-05, Gets “Animal & Animal Products”
  + Commodity Group between 06-15, Gets “Vegetable Products
  + Commodity Group between 16-24, Gets “Foodstuffs”
  + Commodity Group between 25-27, Gets “Mineral Products”
  + Commodity Group between 28-38, Gets “Chemicals & Allied Industries”
  + Commodity Group between 39-40, Gets “Plastics / Rubbers
  + Commodity Group between 41-43, Gets “Raw Hides, Skins, Leather, & Furs”
  + Commodity Group between 44-49, Gets “Wood & Wood Products”
  + Commodity Group between 50-63, Gets “Textiles”
  + Commodity Group between 64-67, Gets “Footwear / Headgear
  + Commodity Group between 68-71, Gets “Stone / Glass” label.
  + Commodity Group between 72-83, Gets “Metals” label.
  + Commodity Group between 84-85, Gets “Machinery / Electrical”
  + Commodity Group between 86-89, Gets “Transportation”

Commodity Group between 90-97, Gets “Miscellaneous” label

# CHAPTER: METHODOLOGY

There are several methodologies for data estimation in the literature. In this study, one of the most innovative ways to make predictions with the help of Artificial Intelligence (AI) technology was used which is named Machine Learning.



## Machine Learning Algorithms

Machine learning, a pivotal field in artificial intelligence (AI) and computer science, revolves around harnessing data and algorithms to mimic human learning processes, progressively upgrading its accuracy. It serves as a vital element in the expanding domain of data science. By employing statistical techniques, algorithms go through training to make classifications, and predictions, and uncover significant insights in data mining endeavors. These valuable insights drive decision-making processes within applications and businesses, aiming to impact key growth metrics positively.

The main objective of this study is to predict the damaged condition of the container under different transportation conditions. The target feature is a Damaged Condition = YES /NO which needs a prediction between two categories and therefore, two class-classification-based ML algorithms are used in the Azure Machine Learning (ML) Studio (Classic) environment (Microsoft Azure, 2021). Microsoft Azure Machine Learning Studio is a powerful tool with several advantages for this study: "Intelligent Container Damage Estimation via Machine Learning Algorithms.". Some key benefits of the Azure machine learning environment can be summarized as follows.

* **User-friendly interface:** The Azure ML Studio provides an intuitive graphical interface that simplifies the machine learning process. Even without extensive coding knowledge, machine learning models can be built, tested, and deployed effortlessly. The drag-and-drop functionality facilitates seamless data preparation, model training, and evaluation, and appeals to users with diverse technical expertise levels.
* **Comprehensive Workflow:** Within Azure ML Studio, a complete platform for the entire machine learning workflow can be found. Everything seamlessly integrates into a single environment, from data import and preprocessing to algorithm selection and configuration, model training, evaluation, and deployment. This cohesive approach streamlines the research process.
* **Extensive Algorithm Library:** Azure ML Studio presents an extensive collection of built-in machine-learning algorithms and modules suitable for this study. Regression, classification, clustering, anomaly detection algorithms, and more exist. The wide array of options empowers data scientists to explore and experiment, ensuring they choose the most appropriate algorithms for the specific research problem.
* **Easy Data Integration:** The tool offers integration with different data sources, which are Azure Blob Storage, SQL databases, Azure Data Lake, and more. This allows easy access and work with the data for model development.
* **Experimentation and Versioning:** Azure Machine Learning Studio enables to execution of model experiments and tracking of the model development steps. Multiple experiments can be generated simultaneously, different models and algorithms can be compared, and model & algorithm performances can be measured. This helps to refine the constructed models iteratively.
* **Scalability and Performance:** With the help of the power of Azure's cloud infrastructure, Azure ML Studio ensures scalability and high-performance capabilities. Cloud-based resources can handle larger datasets and more complex computations and accelerate the training process.
* **Deployment and Integration:** Azure ML Studio offers alternative deployment options for production environments once the machine learning models are developed and trained. Create web services for real-time and on-time scoring and use APIs to construct integrations with other applications.
* **Monitoring and Management:** Azure ML Studio presents monitoring and management features to track the performance and health of generated models. Monitor usage, latency, and other relevant metrics to guarantee your model’s function as targeted. Moreover, the entire workflow can be automated with Azure ML pipelines, simplifying data mining, model deployment, and monitoring.

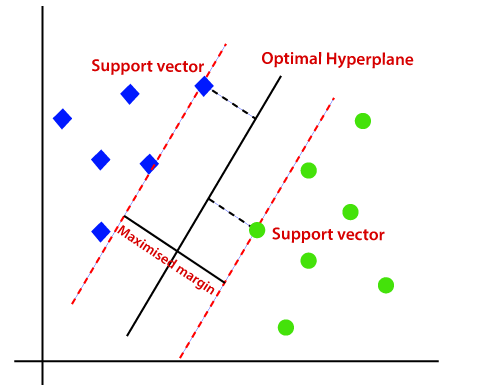
These advantages of Azure Machine Learning Studio make itself a valuable tool for this study, providing a holistic environment to develop, evaluate, and deploy machine learning models.

The six main ML algorithms used for “Two-Class Classification” models in the Azure environment and their brief information and working principles are mentioned below.

### Two-Class Support Vector Machine

Support Vector Machines (SVM) rank among the earliest machine learning algorithms and have found extensive applications in various domains, including information retrieval, text analysis, and image classification. SVMs are multi-purpose and can be utilized for both classification and regression tasks.

As a supervised learning model, SVM needs labeled data during the training. Firstly, SVM analyzes the input data, identifying patterns within a multi-dimensional feature space known as the hyperplane. As indicated in Figure 5.1, each input example is represented as a point in this hyperplane space and is mapped to output categories to accomplish the broadest and most distinct separation between categories. The SVM model targets to create a clear gap between the categories, making easier accurate classification (Microsoft, 2021).



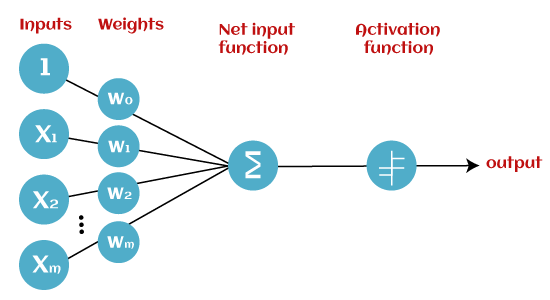
**Figure 5.1.** Working Principle of Two- Class SVM Algorithm.

Source: Javatpoint, n.d.

### Two-Class Average Perceptron

The averaged perceptron method represents a neural network's early and basic variation. This approach includes classifying inputs into various potential outputs through a linear function and then as introduced in Figure 5.2, introduces a set of weights derived from the feature vector, hence the name "perceptron."

Simpler perceptron models are appropriate for learning linearly separable patterns, while neural networks, particularly deep neural networks, can model more complex class boundaries. However, the perceptron model presents faster processing capabilities, and their ability to handle cases in series facilitates them to be used effectively with continuous training. (Microsoft, 2021).



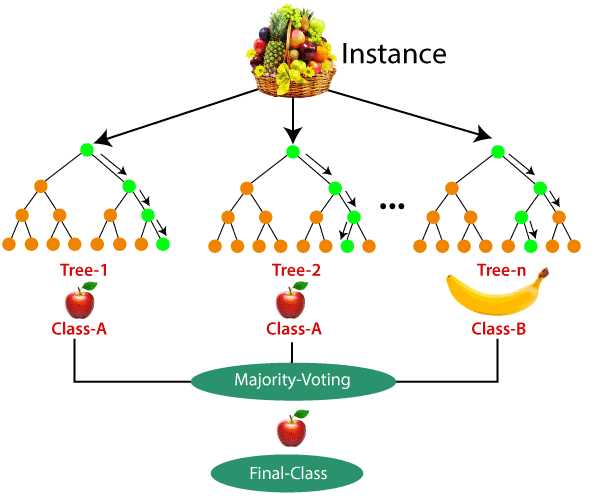
**Figure 5.2.** Working Principle of Two-Class Average Perceptron Algorithm.

Source: Javatpoint, n.d.

### Two-Class Decision Forest

The decision forest algorithm is a powerful ensemble learning method for classification models. Ensemble methods follow the principle of constructing multiple related models instead of a single model. Improved results and a more robust generalization can be achieved with this method. Ensemble models offer more enhanced coverage and accuracy when compared with standalone decision trees.

Multiple techniques exist to create individual models and combine them in an ensemble. Multiple decision trees are built in this two-class decision forest implementation, and the most popular output class is determined entirely via voting as exemplified in Figure 5.3. Voting is a popular approach to generate results in ensemble models, contributing to their effectiveness and reliability.



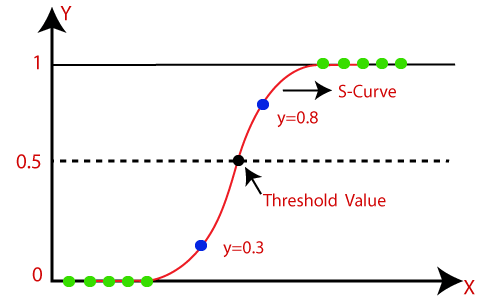
**Figure 5.3.** Working Principle of Two-Class Decision Forest Algorithm.

Source: Javatpoint, n.d.

### Two-Class Logistic Regression

Logistic regression is a widely used and accepted statistical method for predicting an outcome's probability, making it compatible with classification models. The algorithm estimates the likelihood of an event occurring by fitting data to a logistic function (also known as the sigmoid function). The main goal is determining the optimal weights that maximize the observed data's likelihood given the estimated probabilities as presented in Figure 5.4.

Overall, the Two-Class Logistic Regression algorithm models the probability of class membership and provides a flexible approach for binary classification tasks. (Microsoft, 2021)



**Figure 5.4.** Working Principle of Two- Class Logistic Regression Algorithm.

Source: Javatpoint, n.d.

### Two-Class Boosted Decision Tree

The two-class boosted decision tree is an ensemble learning technique where each subsequent tree in the ensemble aims to correct errors made by the preceding tree. Predictions are derived from the joint output of the entire tree ensemble that provides the final forecast.

The algorithm combines multiple decision trees to create a robust ensemble model. During training, the algorithm starts with a simple decision tree and iteratively adds new trees, each attempting to correct the errors made by the previous trees as shown in the graph in Figure 5.5. The process involves assigning higher weights to misclassified samples, emphasizing their importance in subsequent iterations. The algorithm each decision tree's weights and structure to minimize the classification error.

Boosted decision trees, present a simple approach to achieving excellent performance across a diverse range of machine-learning tasks when they are appropriately configured. However, they are memory-intensive learners, and the current implementation holds all information in memory. Consequently, processing large datasets might be challenging for a boosted decision tree model, unlike other machine learning algorithm approaches. (Microsoft, 2021)



**Figure 5.5.** Working Principle of Decision Tree Algorithm.

Source: Javatpoint, n.d.

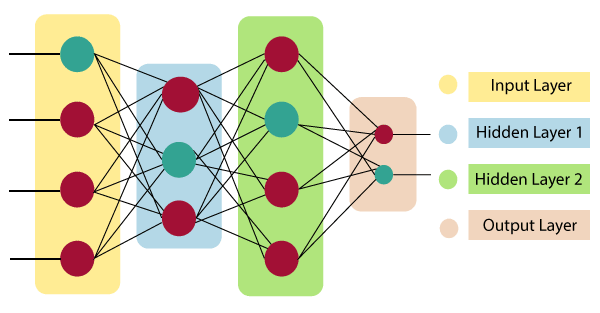
### Two-Class Neural Network

A neural network comprises interconnected layers, where the inputs constitute the first layer and are linked to an output layer through a weighted acyclic graph consisting of nodes and edges. Between the input and output layers, multiple hidden layers can be introduced. While many predictive tasks can be accomplished effectively with only one or a few hidden layers, recent research has demonstrated the effectiveness of deep neural networks (DNN) with numerous layers for complex tasks, such as image or speech recognition. These successive layers are designed to model increasing levels of semantic depth.

The neural network learns the relationship between inputs and outputs by training on the input data. As drawn in Figure 5.6, the graph's direction proceeds from the inputs, through the hidden layers, to the output layer, with all nodes in each layer interconnected by weighted edges to nodes in the subsequent layer. A value is calculated at each node in the hidden and output layers to compute the network's output for a specific input. This value is determined by computing the weighted sum of the values from the nodes in the preceding layer and then applying an activation function to the resulting sum.

Utilizing two-class neural networks for classification involves a supervised learning approach, requiring a labelled dataset with a designated label column. For instance, this neural network model can predict binary outcomes.

Once the model is defined, it undergoes training by feeding it with a tagged dataset through the Train Model input. The trained model can subsequently be used to predict new input data values. (Microsoft, 2021)



**Figure 5.6.** Working Principle of Artificial Neural Network Algorithm.

Source: Javatpoint, n.d.

## Performance Metrics

The main objective of this study is to predict the container damage and risk index. The target feature is a binary value and therefore, regression-based ML algorithms are used in the Azure Machine Learning (ML) Studio environment. Within the content of the Azure ML studio platform, there are different evaluation metrics for the acceptance of the model's validity. (Andrea, 2022):

Firstly, to calculate the performance of the classification-based models, the confusion matrices’ following four main parameters must be calculated as projected in Table 5.1.

* **True Positive:** This value is calculated by finding the model’s “True Positive” predictions while comparing the real occurred observations and it contains examples that have been correctly classified as positive. In this study, if a real happened Damaged = YES observation was accurately predicted by the ML Classification algorithm as Damaged = YES. This performance observation can be counted as a “True Positive”.
* **False Positive:** This value is calculated by finding the model’s “False Positive” predictions while comparing the actual occurred observations. It contains examples incorrectly classified as positive and therefore, actually negative. In this study, if a real occurred Damaged = NO observation was predicted by the ML Classification algorithm correctly as Damaged = YES, then this performance observation can be counted as “False Positive”.
* **True Negative:** This value is calculated by finding the model’s “True Negative” predictions while comparing the real occurred observations and it contains examples that have been correctly classified as negative. In this study, if a real occurred Damaged = NO observation was predicted by the ML Classification algorithm correctly as Damaged = NO, then this performance observation can be counted as “True Negative”.
* **False Negative:** This value is calculated by finding the model’s “False Negative” predictions while comparing the real occurred observations and it contains examples that have been incorrectly classified as negative and are therefore actually positive. In this study, if a real occurred Damaged = YES observation was predicted by the ML Classification algorithm correctly as Damaged = NO, then this performance observation can be counted as “True Negative”.

**Table 5.1.** Confusion Matrix

|  |  |  |  |
| --- | --- | --- | --- |
| **CONFUSION MATRIX** | | **PREDICTED** | |
| **NEGATIVE** | **POSITIVE** |
| **ACTUAL** | **NEGATIVE** | **TRUE NEGATIVES (TN)** | **FALSE POSITIVES (FP)** |
| **POSITIVE** | **FALSE NEGATIVES (FN)** | **TRUE POSITIVES (TP)** |

After constructing the confusion matrix with the main four parameters, the final performance metrics are calculated using these parameters and the other performance metrics which are calculated from these parameters again. The performance metrics and calculation methods of these metrics can be examined below.

### Accuracy

When assessing the effectiveness of a binary classifier, the most frequently used and easily accessible available metric is precise accuracy. This metric calculates the proportion of correctly classified items (True Positives + True Negatives) in the dataset out of the total number of items.

The accuracy is calculated by dividing the number of correct predictions by the total number of predictions the two-class ML model makes. It can be shown below in the formulation (1) (Andrea, 2022):

### Precision

Precision, known as specificity, focuses on the accuracy calculated for positive classes. It quantifies how sensitive the classifier is in recognizing true positives. In other words, precision informs us about the frequency of correct classifications when a class is identified as positive. The precision formula is presented in (2) (Andrea, 2022).

### Recall

If the objective is to maximize the recognition of positive classes, the model must attain a high recall score. The formula (3) of the Recall metric summarizes the situation. In practical terms, it is needed to be considered false negatives instead of false positives. Recall, referred to as sensitivity, describes the phenomenon where, as recall increases, our model becomes less accurate and may classify negative classes as positive.

### F1 Score

As evident from the provided formulas, achieving a model with high precision and high recall is not feasible simultaneously. These two metrics are inversely related: if we increase one, the other must decrease. This phenomenon is known as the precision/recall trade-off. The current scenario makes it evident that utilizing precision or recall as evaluation metrics presents a challenge, as we can only employ one metric at the cost of the other. The F1 score provides a solution to this issue, as it combines precision and recall into a single metric (Andrea, 2022).

The F1 score formulation (4) represents the harmonic mean of precision and recall, and it is widely used as a metric for evaluating binary classification models.

An increase in the F1 score indicates that our model's performance has improved in accuracy, recall, or both. Eventually, the F1 score performance metric will be the most appropriate and effective performance metric for two-class classification ML models. With these advantages, the F1 score metric has been selected as the primary performance metric for this study’s classification-based ML algorithms.

# CHAPTER: COMPUTATIONAL EXPERIMENTS & RESULTS

The intelligent container damage and risk index estimation model has been constructed with six classification-based ML algorithms in Microsoft Azure Machine Learning Studio (Classic) environment (Microsoft Azure, 2021).

The Azure Machine Learning tool provides a user-friendly interface for implementing and experimenting with ML algorithms. More than 15 experiments were run including parameter and setup runs. For each experiment iteration, the best two algorithms are marked according to their F1 score metric percentages. A higher F1 score value indicates the success of the ML model.



## Initial Results of Experiments

The initial experiments include data cleaning & manipulation steps in every iterative run. In each experiment, different data changes were implemented. On the other hand, in initial results, default parameter setups are used which are offered by Azure ML. Additionally, the data split module of the algorithm had been run in default settings which 80 Percent of the data for training, and 20 percent of the data for the test of the model.

Only incorrect data was cleaned from the dataset in the first iteration, and initial results were gathered for the six two-class classification algorithms. Cleaning operations were implemented for the following features: POR haulage, DEL haulage, net weight, and missing container age information.

**Table 6.1.** First Iteration Results of Six Algorithms

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Dataset** | **Incorrect Data Deleted** | | | | | |
| **#1** | **1** | **2** | **3** | **4** | **5** | **6** |
| **Algorithm** | **Two – Class Support Vector Machine** | **Two–Class Averaged Perceptron** | **Two – Class Decision Forest** | **Two – Class Logistic Regression** | **Two – Class Boosted Decision Tree** | **Two–Class Neural Network** |
| **Accuracy** | **0.905** | **0.910** | **0.925** | **0.910** | **0.921** | **0.920** |
| **Precision** | **0.673** | **0.665** | **0.737** | **0.683** | **0.696** | **0.653** |
| **Recall** | **0.225** | **0.665** | **0.466** | **0.296** | **0.463** | **0.538** |
| **F1 Score** | **0.338** | **0.434** | **0.571** | **0.413** | **0.556** | **0.590** |

According to the initial results in Table 6.1, the best two results were gathered from the “Two-Class Decision Forest” and “Two-Class Neural Network” algorithms. Run-time for six algorithms was measured as approximately 2 hours in Azure Cloud resources. The F1 score values are around 0.57 and 0.59. Precision metrics indicate relatively high results.

The second iteration includes outlier and missing data cleaning steps. After the first run, it is determined that some of the data contains missing values in their features. The data which includes missing feature detail is cleaned. In the outlier determination study, the “Net Cargo Weight” feature’s outlier values are cleaned.

**Table 6.2.** Second Iteration Results of Six Algorithms

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Dataset** | **Incorrect Data and Missing Data Deleted** | | | | | |
| **#2** | **1** | **2** | **3** | **4** | **5** | **6** |
| **Algorithm** | **Two – Class Support Vector Machine** | **Two–Class Averaged Perceptron** | **Two – Class Decision Forest** | **Two – Class Logistic Regression** | **Two–Class Boosted Decision Tree** | **Two–Class Neural Network** |
| **Accuracy** | **0.889** | **0.895** | **0.917** | **0.895** | **0.909** | **0.913** |
| **Precision** | **0.617** | **0.637** | **0.740** | **0.636** | **0.675** | **0.789** |
| **Recall** | **0.237** | **0.323** | **0.484** | **0.313** | **0.489** | **0.390** |
| **F1 Score** | **0.342** | **0.429** | **0.585** | **0.419** | **0.587** | **0.522** |

In the second iteration, the best two algorithms are changed marked in Table 6.2, which these are the “Two-Class Decision Forest” and “Two-Class Boosted Decision Tree” algorithms. Run-time again measured around 2 hours. The F1 score values are around 0.585 and 0.587 and their accuracy and precision metrics indicate relatively high results.

The following iterations consist of data manipulation and classification steps. Each subsequent iteration includes different data classification and manipulation operations.

The third, fourth, and fifth iterations include classification processes for the “Net Cargo Weight”, “Container Age”, and “Commodity Group” respectively. The iteration results after the implementation of the classification study are as follows.

**Table 6.3.** Third Iteration Results of Six Algorithms

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Dataset** | **False & Wrong Data and Missing Data Deleted – Net Weight Classification** | | | | | |
| **#3** | **1** | **2** | **3** | **4** | **5** | **6** |
| **Algorithm** | **Two – Class Support Vector Machine** | **Two–Class Averaged Perceptron** | **Two–Class Decision Forest** | **Two – Class Logistic Regression** | **Two–Class Boosted Decision Tree** | **Two–Class Neural Network** |
| **Accuracy** | **0.890** | **0.896** | **0.916** | **0.895** | **0.915** | **0.911** |
| **Precision** | **0.616** | **0.635** | **0.750** | **0.638** | **0.755** | **0.787** |
| **Recall** | **0.240** | **0.340** | **0.466** | **0.317** | **0.490** | **0.366** |
| **F1 Score** | **0.345** | **0.443** | **0.595** | **0.423** | **0.593** | **0.500** |

**Table 6.4.** Fourth Iteration Results of Six Algorithms

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Dataset** | **False & Wrong Data and Missing Data Deleted – Net Weight / Container Age Classification** | | | | | |
| **#4** | **1** | **2** | **3** | **4** | **5** | **6** |
| **Algorithm** | **Two – Class Support Vector Machine** | **Two–Class Averaged Perceptron** | **Two–Class Decision Forest** | **Two – Class Logistic Regression** | **Two – Class Boosted Decision Tree** | **Two – Class Neural Network** |
| **Accuracy** | **0.889** | **0.895** | **0.917** | **0.895** | **0.909** | **0.913** |
| **Precision** | **0.617** | **0.639** | **0.740** | **0.642** | **0.674** | **0.676** |
| **Recall** | **0.246** | **0.327** | **0.495** | **0.315** | **0.487** | **0.555** |
| **F1 Score** | **0.351** | **0.433** | **0.590** | **0.423** | **0.566** | **0.610** |

**Table 6.5.** Fifth Iteration Results of Six Algorithms

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Dataset** | **False & Wrong Data and Missing Data Deleted – Net Weight / Container Age / Commodity Group Classification** | | | | | |
| **#5** | **1** | **2** | **3** | **4** | **5** | **6** |
| **Algorithm** | **Two – Class Support Vector Machine** | **Two–Class Averaged Perceptron** | **Two – Class Decision Forest** | **Two – Class Logistic Regression** | **Two–Class Boosted Decision Tree** | **Two–Class Neural Network** |
| **Accuracy** | **0.875** | **0.895** | **0.922** | **0.895** | **0.909** | **0.924** |
| **Precision** | **0.645** | **0.639** | **0,748** | **0.642** | **0.674** | **0.688** |
| **Recall** | **0.250** | **0.327** | **0.499** | **0.315** | **0.487** | **0.575** |
| **F1 Score** | **0.378** | **0.433** | **0.598** | **0.423** | **0.566** | **0.626** |

Three iteration results summarized in tables 6.3, 6.4, and 6.5 show that the data manipulation and classification step is increasing the model successes slightly. In the fourth iteration, the best two algorithms were updated as “Two-Class Decision Forest” and “Two-Class Neural Network”. The F1 score is measured at 0,598 and 0,626 respectively in the last iteration of the Initial experiments. Moreover, Precision and Recall values seem close to each other, which can be commented that the data becomes balanced. On the other hand, the last iteration’s run time was measured at around one and a half hours which is shorter than the first two iterations. In summary, the model's success is improved in the first five iterations. Additionally, the data manipulation step seems adequate for the model’s success and efficiency in terms of run-time.

## Results After Parameter Setup

After gathering the best result from the initial results, the parameter setup process was applied to increase the model's success. In the parameter setup process, two methods are implemented in the experiments. The first one is the two-class ML model’s parameter optimization. The second one is the Training – Test Ratio splitting decision and setup.

In the first step, two-class ML models’ parameters were optimized. For the optimization, the Azure ML environment’s “Tune Model Hyperparameter” module has been applied with default settings for six selected ML algorithms, as shown in Figure 6.1.

The “Tune Model Hyperparameter” module conducts a parameter sweep, exploring various parameter settings. Doing so identifies the optimal set of hyperparameters, which may vary based on each decision tree, dataset, or regression method. This process of discovering the best configuration is commonly referred to as tuning.

metin, ekran görüntüsü, diyagram, çizgi içeren bir resim

Açıklama otomatik olarak oluşturuldu

**Figure 6.1.** Tune Model Hyperparameter Module in Azure ML (Classic)

Source: Author

After hyperparameter tuning of the algorithms, the below results are generated from the model in Figure 6.2. The results show that hyperparameter tuning increased all models’ performance metrics. Two-Class Decision Forest” and “Two-Class Neural Network” are the best two algorithms. Table 6.6 summarizes their main performance metrics; F1 scores have increased significantly: 0.607 and 0.636 respectively. The run-time of the experiment has not changed. The completion time of the experiment measured around one and a half hours.

beyaz, çizgi, diyagram içeren bir resim

Açıklama otomatik olarak oluşturuldu

**Figure 6.2.** Constructed Final Azure ML Experiment with 6 Algorithms.

Source: Author

**Table 6.6**. Sixth Iteration Results of Six Algorithms

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Dataset** | **Tune Model Hyperparameters for Six Algorithm** | | | | | |
| **#6** | **1** | **2** | **3** | **4** | **5** | **6** |
| **Algorithm** | **Two – Class Support Vector Machine** | **Two – Class Averaged Perceptron** | **Two–Class Decision Forest** | **Two – Class Logistic Regression** | **Two – Class Boosted Decision Tree** | **Two–Class Neural Network** |
| **Accuracy** | **0.899** | **0.901** | **0.930** | **0.904** | **0.912** | **0.939** |
| **Precision** | **0.637** | **0.645** | **0,757** | **0.645** | **0.682** | **0.701** |
| **Recall** | **0.275** | **0.345** | **0.508** | **0.367** | **0.498** | **0.583** |
| **F1 Score** | **0.383** | **0.449** | **0.607** | **0.466** | **0.575** | **0.636** |

The second phase of the experiment's parameter setup is tuning the dataset's split ratio. 80 Training – 20 test ratio is applied for the experiments so far. 80-20 ratio is the default ratio of Azure ML environment’s split module as shown in Figure 6.3 with details. On the other hand, the 80-20 train-test ratio is determined as the optimal ratio in the literature studies.

The Split module has a tunable parameter called “Stratified Split” which takes True / False values. This parameter has critical importance for the used dataset since the usage area of the parameter is when the dataset’s target value is imbalanced because stratified sampling involves dividing the data so that each output dataset receives an approximately equal percentage of each target value.

This study’s target value is to estimate Damage condition = YES / NO. Since the damaged condition is rarer than the non-damaged condition, the number of “NO” s is significantly greater than that “YES” s. Therefore, the stratification parameter was conducted in splitting alternatives.

Moreover, the Split Data module has one more important feature for the generated dataset: Randomized Split. The randomized split option is chosen when it is wished to select data for both groups randomly. This option is beneficial for creating training and test datasets. For this reason, this parameter was taken into consideration in the determination of split data alternatives.

metin, ekran görüntüsü, yazılım, diyagram içeren bir resim

Açıklama otomatik olarak oluşturuldu

**Figure 6.3.** Split Data Module in Azure ML (Classic).

Source: Author

Therefore, for determining the data splitting train-test ratio and its pre-determined two parameter combinations, the static design of the experiment study was applied, and the following alternatives are defined in Table 6.7.

**Table 6.7.** Split Data Module Experiment Alternatives via Static Design of Experiment

|  |  |  |  |
| --- | --- | --- | --- |
| **Split Ratio (%)** | | **Stratified Split (YES / NO)** | **Randomized Split** |
| **Train** | **Test** |
| 70 | 30 | NO | YES |
| 70 | 30 | YES | YES |
| 75 | 25 | NO | YES |
| 75 | 25 | YES | YES |
| 80 | 20 | NO | YES |
| 80 | 20 | YES | YES |
| 80 | 20 | YES | NO |
| 85 | 15 | NO | YES |
| 85 | 15 | YES | YES |
| 90 | 10 | NO | YES |
| 90 | 10 | YES | YES |

All designed split data alternatives were conducted in iterative experiments. After eleven iterations, the best-determined result was calculated as follows. Other running options have been shared in the appendices section.

The best result was gathered from Split Ratio: 80 train – 20 test, Randomized Split = YES, Stratified Split = YES parameter combination summarized in table 6.8. With this Split Data combination, all six models are better trained when compared with the standardized 80-train – 20-train split data method. The best two algorithms were selected: “Two-Class Decision Forest” and “Two-Class Neural Network” with this tuning setup. Since this iteration is the best-resulted experiment between all conducted experiment sets, it can be concluded that the “Two-Class Neural Network” algorithm was given the best result with an F1 score value of 0.714 and other performance metrics showed significant improvement.

**Table 6.8.** Best Experiment Result Between a Total of 17 Experiments

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Dataset** | **Split Ratio: 80-20 / Randomized Split = YES /** | | | | | |
| **Stratified Split = YES** | | | | | |
| **#9** | **1** | **2** | **3** | **4** | **5** | **6** |
| **Algorithm** | **Two - Class Support Vector Machine** | **Two - Class Averaged Perceptron** | **Two - Class Decision Forest** | **Two - Class Logistic Regression** | **Two - Class Boosted Decision Tree** | **Two - Class Neural Network** |
| **Accuracy** | **0.910** | **0.955** | **0.940** | **0.907** | **0.906** | **0.943** |
| **Precision** | **0.655** | **0.688** | **0,796** | **0.659** | **0.702** | **0.784** |
| **Recall** | **0.302** | **0.369** | **0.556** | **0.373** | **0.519** | **0.657** |
| **F1 Score** | **0.413** | **0.480** | **0.654** | **0.476** | **0.596** | **0.714** |

## Discussion of Results

The experiment’s highest main performance metric F1 score value was 0.590 in the first iteration. In the initial running phase, implemented data cleaning and manipulation steps increased the model F1 score value to 0.626. In addition, Accuracy, Precision, and Recall metrics also increased effectively. After applying the two-phase parameter setup, the F1 score value reached around 0.714. The summary of the three primary iterations and their results was shared in Table 6.9.

**Table 6.9**. Summary Table of Main Iterations and Results

|  |  |  |  |
| --- | --- | --- | --- |
| **Iterations** | **Initial Run** | **Data Cleaning & Manipulation & Classification** | **Parameter Setup (Model + Train - Test Data Splitting)** |
| **Algorithm** | **Two–Class Neural Network** | **Two–Class Neural Network** | **Two - Class Neural Network** |
| **Accuracy** | **0.920** | **0.924** | **0.943** |
| **Precision** | **0.653** | **0.688** | **0.784** |
| **Recall** | **0.538** | **0.575** | **0.657** |
| **F1 Score** | **0.590** | **0.626** | **0.714** |

A total of 17 experiments were executed and the best algorithm: “Two-Class Neural Network” was determined with an Accuracy value of 0.943, Precision value of 0.784, Recall value of 0.657, and F1 score measured of 0.714. These performance metric results highlighted that the model estimation power is acceptable for the usage of the model.

Azure ML environment provides a web service (API) solution to call the generated experiments and their models from other applications. Therefore, this model can be integrated and called from an ERP with this method. The constructed web service can also be tested from an Azure ML user interface as shown below in Figure 6.4. All it needs is, to enter parameters and get the results as added in Figure 6.5 according to the entered input features.

metin, ekran görüntüsü, sayı, numara, yazı tipi içeren bir resim

Açıklama otomatik olarak oluşturuldu

**Figure 6.4.** Azure ML Webservice Test Interface for Entering Feature Inputs.

Source: Author

metin, yazı tipi, yazılım, web sayfası içeren bir resim

Açıklama otomatik olarak oluşturuldu

**Figure 6.5.** Azure ML Webservice Test Result Output.

Source: Author

# CHAPTER: CONCLUSIONS AND FUTURE WORK

In conclusion, under the presented application of “An Intelligent Container Damage Estimation Tool,” it is proved that the ML approach can be applied to container damage estimations.

The target feature is a binary variable type, which is why this experiment uses six two-class classification-based algorithms within the Microsoft Azure Machine Learning Studio. The data collection, cleansing, and manipulation steps are completed for proper preparation of the features as input, and feature classifications are developed.

The prepared data set was run on Azure ML environment with six algorithms: “Two-Class Support Vector Machine”, “Two-Class Averaged Perceptron”, “Two-Class Decision Forest”, “Two-Class Logistic Regression”, “Two-Class Boosted Decision Tree”, “Two-Class Neural Network”. Precision, recall, and F1 score performance metrics were generated from the Confusion Matrix to measure the success of the model accuracy. The primary performance metric had been accepted as the F1 score because of its advantages. In the initial experimentation phase, the iteration results are collected for the cleaned and manipulated data, the F1 score value is increased from 0.590 to 0.629 by approximately a 7% increase, and run-time efficiency has been increased. Followingly, the two-phase parameter setup operation increased the F1 score value to 0.714 by about 14% increase and the best results were taken from the “Two–Class Neural Network” algorithm.

After improving the performance measure values, the model has been generated as a web service using Azure ML environment’s quick sharing feature to call algorithms from external applications with feature parameters. The model was tested from the feature input test interface of the Azure ML as can be seen in Figure 6.4 and Figure 6.5.

Moreover, as mentioned in the Methodology section, the Azure ML environment has many advantages during the construction phase of the model and especially in the maintenance phase of the model. Making changes to the model has become easier with the Azure ML environment when the model is in active usage.

As a result, it can be summarized that implementing classification-based ML algorithms can be advantageous for estimating container-damaged situations. This model's prediction capability empowers the business responsible for appropriate container selection decisions for that transportation. By entering input features of the model before assigning a container for that transportation, the output prediction is going to be beneficial for taking container damage risks. According to the output of this prediction, the related business responsible can decide about the proper container selection. For example, model output indicates potential damage to the container, the responsible can select the canceling of the transportation or choose a more expendable or old container to avoid risks. On the other hand, the damaged situation of the container is generating repair costs for the liner shipping companies. Also, opportunity cost occurs while the container is in the repair process.

The algorithm’s performance metrics can be increased by evaluating and applying more collectable input features in future work. Moreover, parameter setup is also critical for performance metric increment. Implementing custom parameter setup experiment iterations can widen the parameter setup study.

After estimating the container's damaged state with accurate and acceptable performance levels, the model can be called from an ERP with the help of application programming interface (API) technology. With the integration of the model into the ERP application, the damage situation can be estimated in real-time. All feature parameters will get from the ERP and after the selection of the container, selection, the system will start the ML model to predict the damage situation. After getting the damaged condition of the container in advance, the proper container can be selected from the available stocks of the system by integrating a basic container advisor system into ERP.

Followingly, by using some parameters like damage risk ratio which is the primary performance metric of the model, the “Container Damage Risk Surcharge or Fee” will be calculated for minimizing damage-related costs by using damage information in advance and can calculated fee may be charged to the customers from an ERP system. For this calculation, insurance risk premium calculation methods can be implemented to minimize the weight of repair & scrap costs of the containers on the container owner companies.

# REFERENCES

Andrea, D. (2022). *The Explanation You Need on Binary Classification Metrics*. Towards Data Science. Retrieved February 19, 2023, from https://towardsdatascience.com/the-explanation-you-need-on-binary-classification-metrics-321d280b590f

Assiri, A. (2020). Anomaly Classification Using Genetic Algorithm-Based Random Forest Model for Network Attack Detection. *Tech Science Pass, Computers, Materials & Continua*, *66(1),* 767-768. <https://doi.org/10.32604/cmc.2020.013813>

Caruana, R., Lawrence, S., & Giles, L. (2000). Overfitting in Neural Nets: Backpropagation, Conjugate Gradient, and Early Stopping. *DBLP, Advances in Neural Information Processing Systems,* *13*, 402-408

Chang, C. H., & Xu, J. (2015). Risk analysis for container shipping: from a logistics perspective. The International Journal of Logistics Management, 26(1), 147 – 171. <https://doi.org/10.1108/IJLM-07-2012-0068>

Cogoport. (2021). HS Code: Classification of Goods In Export-Import. Retrieved February 20, 2023, from https://www.cogoport.com/en-IN/blogs/hs-code-all-about-classification-of-goods-in-export-import

Ellis, J. (2011). Analysis of accidents and incidents occurring during transport of packaged dangerous goods by sea*. Elsevier, Safety Science,* *49*, 1231–1237, <https://doi.org/10.1016/j.ssci.2011.04.004>

Fahad, L. G., & Tahir, S. F. (2020). Activity recognition and anomaly detection in smart homes. *Elsevier, Neurocomputing*, *423*, 362–372, <https://doi.org/10.1016/j.neucom.2020.10.102>

Feng, Y., Wang, Q., Wu, D., Gao, W., & Tin-Loi, F. (2020). Stochastic nonlocal damage analysis by a machine learning approach. *Elsevier, Computer Methods in Applied Mechanics and Engineering,* *372,* 113371

Foreign Trade. (n.d.). Harmonized System Codes (HS Code 2017 - Current). Retrieved February 20, 2023, from https://www.foreign-trade.com/reference/hscode.htm

Hart, E., Sim, K., & Kamimura, K. (2019). Use of machine learning techniques to model wind damage to forests. *Elsevier, Agricultural and Forest Meteorology, 265*, 16–29. <https://doi.org/10.1016/j.agrformet.2018.10.022>

Imamoglu, Z., Tuglular, T., & Bastanlar, Y. (2020). Container Damage Detection and Classification Using Container Images. *IEEE, 2020 28th Signal Processing and Communications Applications Conference (SIU),* 1-4. <https://doi.org/10.1109/SIU49456.2020.9302442>

International Trade Administration. (n.d.). Harmonized System (HS) Codes. Retrieved February 20, 2023, from https://www.trade.gov/harmonized-system-hs-codes

Javatpoint. (n.d.). *Artificial Neural Network Tutorial*. Retrieved September 15, 2022, from

https://www.javatpoint.com/artificial-neural-network

Javatpoint. (n.d.). *Decision Tree Classification Algorithm*. Retrieved September 9, 2022, from

https://www.javatpoint.com/machine-learning-decision-tree-classification-algorithm

Javatpoint. (n.d.). Logistic Regression in Machine Learning. Retrieved September 7, 2022, from https://www.javatpoint.com/logistic-regression-in-machine-learning

Javatpoint. (n.d.). *Perceptron in Machine Learning*. Retrieved September 3, 2022, from

https://www.javatpoint.com/perceptron-in-machine-learning

Javatpoint. (n.d.). *Random Forest Algorithm*. Retrieved September 4, 2022, from

https://www.javatpoint.com/machine-learning-random-forest-algorithm

Javatpoint. (n.d.). *Support Vector Machine Algorithm*. Retrieved September 2, 2022, from

https://www.javatpoint.com/machine-learning-support-vector-machine-algorithm

Knapp, S. and Heij, C. (2017). Evaluation of total risk exposure and insurance premiums in the maritime industry. *Elsevier, Transportation Research Part D*, 54, 321–334. https://doi.org/[10.1016/j.trd.2017.06.001](https://doi.org/10.1016/j.trd.2017.06.001)

Kulkarni, H., Kumbham, B., & Mani, J. (2018). Multiclass Classification to Predict the Level of Storm and Damages Using Support Vector Machine. *IEEE, 2018 Fourteenth International Conference on Information Processing (ICINPRO),* 1-5. <https://doi.org/10.1109/ICINPRO43533.2018.9096705>

Liu, Y., Pang, Z., Karlsson, M., & Gong, S. (2020). Anomaly detection based on machine learning in IoT-based vertical plant wall for indoor climate control. *Elsevier, Building and Environment*, *183*, 107212

Microsoft Azure. (2021). *Azure Machine Learning*. Retrieved August 15, 2022, from

<https://azure.microsoft.com/en-us/services/machine-learning/>

Microsoft. (2021). Two-Class Average Perceptron component. Retrieved September 3, 2022, from https://learn.microsoft.com/en-us/azure/machine-learning/component-reference/two-class-averaged-perceptron?view=azureml-api-2

Microsoft. (2021). *Two-Class Boosted Decision Tree component*. Retrieved September 9, 2022, from https://learn.microsoft.com/en-us/azure/machine-learning/component-reference/two-class-boosted-decision-tree?view=azureml-api-2

Microsoft. (2021). *Two-Class Decision Forest component.* Retrieved September 4, 2022, from

https://learn.microsoft.com/en-us/azure/machine-learning/component-reference/two-class-decision-forest?view=azureml-api-2

Microsoft. (2021). *Two-Class Logistic Regression component*. Retrieved September 7, 2022, from https://learn.microsoft.com/en-us/azure/machine-learning/component-reference/two-class-logistic-regression?view=azureml-api-2

Microsoft. (2021). *Two-Class Neural Network component*. Retrieved September 15, 2022, from https://learn.microsoft.com/en-us/azure/machine-learning/component-reference/two-class-neural-network?view=azureml-api-2

Microsoft. (2021). Two-Class Support Vector Machine component. Retrieved September 2, 2022, from https://learn.microsoft.com/en-us/azure/machine-learning/component-reference/two-class-support-vector-machine?view=azureml-api-2

Nguyen, S., Chen, P. S. L., Du, Y., & Shi, W. (2019). A quantitative risk analysis model with integrated deliberative Delphi platform for container shipping operational risks. *Elsevier, Transportation Research Part E*, *129,* 203–227. [https://doi.org/10.1016/j.tre.2019.08.002](https://doi.org/10.1016/j.tre.2019.08.002%20)

Patil, K., Kulkarni, M., Sriraman, A., & Karande, S. (2017). Deep Learning Based Car Damage Classification. *IEEE, 2017 16th IEEE International Conference on Machine Learning and Applications,* 50-54. <https://doi.org/10.1109/ICMLA.2017.0-179>

Quest. (2023). *Toad for Oracle - Technical Documentation.* Retrieved January 19, 2023, from

https://support.quest.com/toad-for-oracle/16.1/technical-documents

Ray, S. (2019). A Quick Review of Machine Learning Algorithms, *IEEE, 2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing* (COMITCon), 35-39. <https://doi.org/10.1109/COMITCon.2019.8862451>

Shang, K.C., & Tseng, W. J. (2010). A Risk Analysis of stevedoring operations in Seaport container terminals. *Journal of Marine Science and Technology, 18(2),* 201-210. <https://doi.org/10.51400/2709-6998.2319>

Singh, S. P., Antle, J., Singh, J., Topper, E., & Grewal, G. (2014). Load Securement and Packaging Methods to Reduce Risk of Damage and Personal Injury for Cargo Freight in Truck, Container, and Intermodal Shipments. *The Journal of Applied Packaging Research, 6(1),* 47-62. https://doi.org/[10.14448/japr.01.0005](http://dx.doi.org/10.14448/japr.01.0005)

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Semantic, Journal of Machine Learning Research*. *15*, 1929-1958

Tseng, W. J., Ding, J., & Li, M. (2013). Risk management of cargo damage in export operations of ocean freight forwarders in Taiwan. *Journal of Engineering for the Maritime Environment*, *229(3),* 232–247. <https://doi.org/10.1177/1475090213513755>

Unctad. (2019). *Review of Maritime Transport 2019.* Retrieved December 8, 2021, from<https://unctad.org/webflyer/review-maritime-transport-2019>

Wan, C., Yan, X., Zhang, D., Qu, Z., & Yang, Z. (2019). An advanced fuzzy Bayesian-based FMEA approach for assessing maritime supply chain risks. *Elsevier, Transportation Research Part E, 12,* 222–240. <https://doi.org/10.1016/j.tre.2019.03.011>

Wei, J., Chang, L., & Jasmine, S. L. (2019). Cyclone risk model and assessment for East Asian container ports. *Elsevier Ocean and Coastal Management*, 178, 104796. <https://doi.org/10.1016/j.ocecoaman.2019.04.023>

Wei, J., Huang, H., Yao, L., Hu, Y., Fan, Q., & Huang, D. (2020). New imbalanced bearing fault diagnosis method based on Sample-characteristic Oversampling TechniquE (SCOTE) and multi-class LS-SVM. *Elsevier, Applied Soft Computing Journal, 101*, 107043

# APPENDICES

## APPENDIX 1. RESULTS OF THE EXPERIMENTS

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Dataset** | **Split Ratio: 80-20 / Randomized Split = YES /** | | | | | |
| **Stratified Split = NO** | | | | | |
| **#7** | **1** | **2** | **3** | **4** | **5** | **6** |
| **Algorithm** | **Two - Class Support Vector Machine** | **Two - Class Averaged Perceptron** | **Two - Class Decision Forest** | **Two - Class Logistic Regression** | **Two - Class Boosted Decision Tree** | **Two - Class Neural Network** |
| **Accuracy** | **0.899** | **0.901** | **0.930** | **0.904** | **0.912** | **0.939** |
| **Precision** | **0.637** | **0.645** | **0,757** | **0.645** | **0.682** | **0.701** |
| **Recall** | **0.275** | **0.345** | **0.508** | **0.367** | **0.498** | **0.583** |
| **F1 Score** | **0.383** | **0.449** | **0.607** | **0.466** | **0.575** | **0.626** |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
| **Dataset** | **Split Ratio: 80-20 / Randomized Split = NO /** | | | | | |
| **Stratified Split = NO** | | | | | |
| **#8** | **1** | **2** | **3** | **4** | **5** | **6** |
| **Algorithm** | **Two - Class Support Vector Machine** | **Two - Class Averaged Perceptron** | **Two - Class Decision Forest** | **Two - Class Logistic Regression** | **Two - Class Boosted Decision Tree** | **Two - Class Neural Network** |
| **Accuracy** | **0.878** | **0.878** | **0.817** | **0.875** | **0.878** | **0.122** |
| **Precision** | **1.000** | **1.000** | **0.346** | **0.482** | **1.000** | **0.122** |
| **Recall** | **0.000** | **0.000** | **0.567** | **0.350** | **0.000** | **1000** |
| **F1 Score** | **0.000** | **0.000** | **0.430** | **0.406** | **0.000** | **0.217** |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
| **Dataset** | **Split Ratio: 80-20 / Randomized Split = YES /** | | | | | |
| **Stratified Split = YES** | | | | | |
| **#9** | **1** | **2** | **3** | **4** | **5** | **6** |
| **Algorithm** | **Two - Class Support Vector Machine** | **Two - Class Averaged Perceptron** | **Two - Class Decision Forest** | **Two - Class Logistic Regression** | **Two - Class Boosted Decision Tree** | **Two - Class Neural Network** |
| **Accuracy** | **0.910** | **0.955** | **0.940** | **0.907** | **0.906** | **0.943** |
| **Precision** | **0.655** | **0.688** | **0,796** | **0.659** | **0.702** | **0.784** |
| **Recall** | **0.302** | **0.369** | **0.556** | **0.373** | **0.519** | **0.657** |
| **F1 Score** | **0.413** | **0.480** | **0.654** | **0.476** | **0.596** | **0.714** |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
| **Dataset** | **Split Ratio: 75-25 / Randomized Split = YES /** | | | | | |
| **Stratified Split = NO** | | | | | |
| **#10** | **1** | **2** | **3** | **4** | **5** | **6** |
| **Algorithm** | **Two - Class Support Vector Machine** | **Two - Class Averaged Perceptron** | **Two - Class Decision Forest** | **Two - Class Logistic Regression** | **Two - Class Boosted Decision Tree** | **Two - Class Neural Network** |
| **Accuracy** | **0.889** | **0.895** | **0.916** | **0.895** | **0.909** | **0.915** |
| **Precision** | **0.614** | **0.637** | **0.736** | **0.637** | **0.674** | **0.727** |
| **Recall** | **0.246** | **0.321** | **0.484** | **0.317** | **0.488** | **0.485** |
| **F1 Score** | **0.352** | **0.427** | **0.584** | **0.423** | **0.566** | **0.582** |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
| **Dataset** | **Split Ratio: 75-25 / Randomized Split = YES /** | | | | | |
| **Stratified Split = YES** | | | | | |
| **#11** | **1** | **2** | **3** | **4** | **5** | **6** |
| **Algorithm** | **Two - Class Support Vector Machine** | **Two - Class Averaged Perceptron** | **Two - Class Decision Forest** | **Two - Class Logistic Regression** | **Two - Class Boosted Decision Tree** | **Two - Class Neural Network** |
| **Accuracy** | **0.888** | **0.895** | **0.916** | **0.895** | **0.909** | **0.915** |
| **Precision** | **0.612** | **0.636** | **0.737** | **0.642** | **0.676** | **0.705** |
| **Recall** | **0.230** | **0.333** | **0.484** | **0.317** | **0.488** | **0.516** |
| **F1 Score** | **0.335** | **0.437** | **0.585** | **0.425** | **0.567** | **0.596** |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
| **Dataset** | **Split Ratio: 85-15 / Randomized Split = YES /** | | | | | |
| **Stratified Split = NO** | | | | | |
| **#12** | **1** | **2** | **3** | **4** | **5** | **6** |
| **Algorithm** | **Two - Class Support Vector Machine** | **Two - Class Averaged Perceptron** | **Two - Class Decision Forest** | **Two - Class Logistic Regression** | **Two - Class Boosted Decision Tree** | **Two - Class Neural Network** |
| **Accuracy** | **0.890** | **0.895** | **0.916** | **0.895** | **0.908** | **0.914** |
| **Precision** | **0.623** | **0.641** | **0.743** | **0.643** | **0.671** | **0.751** |
| **Recall** | **0.238** | **0.311** | **0.477** | **0.316** | **0.488** | **0.447** |
| **F1 Score** | **0.345** | **0.419** | **0.581** | **0.424** | **0.565** | **0.561** |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
| **Dataset** | **Split Ratio: 85-15 / Randomized Split = YES /** | | | | | |
| **Stratified Split = YES** | | | | | |
| **#13** | **1** | **2** | **3** | **4** | **5** | **6** |
| **Algorithm** | **Two - Class Support Vector Machine** | **Two - Class Averaged Perceptron** | **Two - Class Decision Forest** | **Two - Class Logistic Regression** | **Two - Class Boosted Decision Tree** | **Two - Class Neural Network** |
| **Accuracy** | **0.889** | **0.895** | **0.916** | **0.895** | **0.909** | **0.913** |
| **Precision** | **0.612** | **0.633** | **0.733** | **0.638** | **0.680** | **0.672** |
| **Recall** | **0.242** | **0.338** | **0.488** | **0.318** | **0.482** | **0.563** |
| **F1 Score** | **0.347** | **0.441** | **0.586** | **0.424** | **0.564** | **0.612** |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
| **Dataset** | **Split Ratio: 90-10 / Randomized Split = YES /** | | | | | |
| **Stratified Split = NO** | | | | | |
| **#14** | **1** | **2** | **3** | **4** | **5** | **6** |
| **Algorithm** | **Two - Class Support Vector Machine** | **Two - Class Averaged Perceptron** | **Two - Class Decision Forest** | **Two - Class Logistic Regression** | **Two - Class Boosted Decision Tree** | **Two - Class Neural Network** |
| **Accuracy** | **0.890** | **0.896** | **0.916** | **0.895** | **0.908** | **0.915** |
| **Precision** | **0.623** | **0.637** | **0.736** | **0.641** | **0.672** | **0.762** |
| **Recall** | **0.235** | **0.331** | **0.488** | **0.314** | **0.484** | **0.433** |
| **F1 Score** | **0.341** | **0.436** | **0.586** | **0.422** | **0.563** | **0.552** |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
| **Dataset** | **Split Ratio: 90-10 / Randomized Split = YES /** | | | | | |
| **Stratified Split = YES** | | | | | |
| **#15** | **1** | **2** | **3** | **4** | **5** | **6** |
| **Algorithm** | **Two - Class Support Vector Machine** | **Two - Class Averaged Perceptron** | **Two - Class Decision Forest** | **Two - Class Logistic Regression** | **Two - Class Boosted Decision Tree** | **Two - Class Neural Network** |
| **Accuracy** | **0.888** | **0.895** | **0.916** | **0.894** | **0.909** | **0.910** |
| **Precision** | **0.607** | **0.638** | **0.739** | **0.634** | **0.672** | **0.832** |
| **Recall** | **0.235** | **0.326** | **0.484** | **0.317** | **0.487** | **0.325** |
| **F1 Score** | **0.339** | **0.431** | **0.585** | **0.423** | **0.565** | **0.467** |
|  |  |  |  |  |  |  |
| **Dataset** | **Split Ratio: 70-30 / Randomized Split = YES /** | | | | | |
| **Stratified Split = NO** | | | | | |
| **#16** | **1** | **2** | **3** | **4** | **5** | **6** |
| **Algorithm** | **Two - Class Support Vector Machine** | **Two - Class Averaged Perceptron** | **Two - Class Decision Forest** | **Two - Class Logistic Regression** | **Two - Class Boosted Decision Tree** | **Two - Class Neural Network** |
| **Accuracy** | **0.889** | **0.895** | **0.916** | **0.895** | **0.909** | **0.912** |
| **Precision** | **0.612** | **0.630** | **0.739** | **0.639** | **0.673** | **0.758** |
| **Recall** | **0.240** | **0.340** | **0.479** | **0.317** | **0.489** | **0.409** |
| **F1 Score** | **0.345** | **0.442** | **0.581** | **0.424** | **0.566** | **0.531** |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
| **Dataset** | **Split Ratio: 70-30 / Randomized Split = YES /** | | | | | |
| **Stratified Split = YES** | | | | | |
| **#17** | **1** | **2** | **3** | **4** | **5** | **6** |
| **Algorithm** | **Two - Class Support Vector Machine** | **Two - Class Averaged Perceptron** | **Two - Class Decision Forest** | **Two - Class Logistic Regression** | **Two - Class Boosted Decision Tree** | **Two - Class Neural Network** |
| **Accuracy** | **0.889** | **0.895** | **0.916** | **0.895** | **0.909** | **0.910** |
| **Precision** | **0.612** | **0.633** | **0.737** | **0.641** | **0.677** | **0.803** |
| **Recall** | **0.234** | **0.336** | **0.486** | **0.316** | **0.487** | **0.346** |
| **F1 Score** | **0.339** | **0.439** | **0.586** | **0.423** | **0.566** | **0.484** |

## APPENDIX 2. SQL QUERY FOR GATHERING BIG DATA

SELECT SUBSTR(SLDT,1,4) AS SAIL\_YEAR,

CASE SUBSTR(SLDT,5,2)

WHEN '01' THEN 'JANUARY'

WHEN '02' THEN 'FEBRUARY'

WHEN '03' THEN 'MARCH'

WHEN '04' THEN 'APRIL'

WHEN '05' THEN 'MAY'

WHEN '06' THEN 'JUNE'

WHEN '07' THEN 'JULY'

WHEN '08' THEN 'AUGUST'

WHEN '09' THEN 'SEPTEMBER'

WHEN '10' THEN 'OCTOBER'

WHEN '11' THEN 'NOVEMBER'

ELSE 'DECEMBER'

END AS SAIL\_MONTH,

CASE WHEN ORIGIN\_HAUL\_YN = 'C' THEN 'YES' ELSE 'NO' END AS POR\_HAULAGE,

CASE WHEN DEST\_HAUL\_YN = 'C' THEN 'YES' ELSE 'NO' END AS DEL\_HAULAGE,

SUBSTR(MPOL,1,2) AS POL\_COUNTRY,

SUBSTR(MPOD,1,2) AS POD\_COUNTRY,

CASE WHEN TSHIPPORT3 IS NOT NULL THEN 3

WHEN TSHIPPORT3 IS NULL AND TSHIPPORT2 IS NOT NULL THEN 2

WHEN TSHIPPORT2 IS NULL AND TSHIPPORT1 IS NOT NULL THEN 1

WHEN TSHIPPORT1 IS NULL THEN 0

END AS NUMBER\_OF\_TRANSSHIPMENT,

MVES AS VESSEL, CUR\_SRV AS SERVICE,

DP01.EQSZ AS EQP\_SIZE, DP01.EQTP AS EQP\_TYPE, DP01.MT\_FL\_FLAG AS EMPTY\_FULL\_FLAG,

CASE WHEN NET\_WEIGHT = 0 THEN 'EMPTY'

WHEN NET\_WEIGHT BETWEEN 1 AND 10000 THEN 'LIGHT'

WHEN NET\_WEIGHT BETWEEN 10001 AND 20000 THEN 'MEDIUM'

ELSE 'HEAVY'

END AS NET\_WEIGHT\_CLASS,

NET\_WEIGHT,

CASE WHEN SUBSTR(SLDT,1,4)-SUBSTR(CP01.MAN\_DATE,1,4) BETWEEN 0 AND 5 THEN '0-5'

WHEN 2020-SUBSTR(CP01.MAN\_DATE,1,4) BETWEEN 6 AND 10 THEN '6-10'

WHEN 2020-SUBSTR(CP01.MAN\_DATE,1,4) BETWEEN 11 AND 15 THEN '11-15'

ELSE '>15'

END AS EQP\_AGE,

2020-SUBSTR(CP01.MAN\_DATE,1,4) AS AGE,

CASE WHEN SUBSTR(COMMODITY.COMMODITY\_CODE,1,2)='E0' OR COMMODITY.COMMODITY\_CODE IS NULL THEN 'EMPTY'

WHEN SUBSTR(COMMODITY.COMMODITY\_CODE,1,2) BETWEEN '01' AND '05' THEN 'ANIMAL PRODUCTS'

WHEN SUBSTR(COMMODITY.COMMODITY\_CODE,1,2) BETWEEN '06' AND '24' THEN 'VEGETABLE AND FOODSTUFFS'

WHEN SUBSTR(COMMODITY.COMMODITY\_CODE,1,2) BETWEEN '25' AND '40' THEN 'CHEMICALS'

WHEN SUBSTR(COMMODITY.COMMODITY\_CODE,1,2) BETWEEN '41' AND '49' THEN 'RAW-WOODS'

WHEN SUBSTR(COMMODITY.COMMODITY\_CODE,1,2) BETWEEN '50' AND '67' THEN 'TEXTILE'

WHEN SUBSTR(COMMODITY.COMMODITY\_CODE,1,2) BETWEEN '68' AND '85' THEN 'VEGETABLE AND FOODSTUFFS'

ELSE 'MISCELLANEOUS'

END AS COMMODITY\_GROUP,

PREV\_WDMG.NOF\_PREV\_WDMG AS PREVIOUS\_DMG\_COUNT,

PREV\_DFUL.NOF\_PREV\_DFUL AS PREVIOUS\_FULL\_LOADS,

PCK.PACK\_AREA AS PACKING\_PLACE,

UNPCK.UNPACK\_AREA AS UNPACKING\_PLACE,

BLNO AS BL\_NO, EQNO AS EQP\_NO,

CASE WHEN (IS\_DMG.DMG\_FLAG != '1' OR IS\_DMG.DMG\_FLAG IS NULL) THEN 'NO'

ELSE 'YES'

END AS IS\_DAMAGED

FROM DP01,

( SELECT \*

FROM TP63

WHERE

AND SLDT BETWEEN 20200101 AND 20201231

) TP63,

( SELECT BLNO, EQSZ, EQTP, EQNO, MT\_FLL\_FLAG, TARE, SUM(MTWT) AS NET\_WEIGHT

FROM DP55

WHERE (PART\_OF\_FLAG = 'N' OR PART\_OF\_FLAG IS NULL)

AND (SHSH != 1 OR SHSH IS NULL)

GROUP BY BLNO, EQSZ, EQTP, EQNO, MT\_FLL\_FLAG, TARE

) DP55,

CP62,

( SELECT RS2.\*, CODE AS COMMODITY\_CODE, DESC AS COMMODITY\_NAME

FROM ( SELECT MAX\_COMM.\*, RS1.CRSQ, RS1.CMCD,

ROW\_NUMBER() OVER (PARTITION BY MAX\_COMM.BLNO ORDER BY MAX\_COMM.BLNO, RS1.CMCD) AS ROW\_NO

FROM ( SELECT BLNO AS BLNO, MAX(TOT\_PCK) MAX\_COMM

FROM ( SELECT BLNO, CMCD, SUM(PCK) TOT\_PCKG

FROM DP55,

DP50

WHERE BLNO = BLNO

AND CMSQ = CMSQ

GROUP BY BLNO, CMCD

)

GROUP BY BLNO

) MAX\_COMM,

( SELECT BLNO, CRSQ, CMCD, SUM(BYPCK) TOT\_PCKG

FROM DP55,

DP50

WHERE BLNO = BLNO

AND CMSQ = CMSQ

GROUP BY BLNO, CRSQ, CMCD

) RS1

WHERE MAX\_COMM.BLNO = RS1.BLNO

AND MAX\_COMM.MAX\_COMM = RS1.TOT\_PCKG

) RS2,

TP81

WHERE RS2.CMCD = ITP081.CODE(+)

AND ROW\_NO = 1

) COMMODITY,

( SELECT RS1.EQN, SUBSTR(CP03.BOLN,1,16) AS BOLN, 1 AS DMG\_FLAG

FROM ( SELECT CP03.EQN, DMG.SEQN AS DMG\_SEQ, MAX(CP30.SEQN) AS BLED\_SEQ

FROM ( SELECT EQN, SEQN

FROM CP03

WHERE BOLN IS NOT NULL

) CP03,

( SELECT EQN, SEQN

FROM CP03

WHERE MVDT >= 20200101

AND STA = 'DMG'

) DMG

WHERE DMG.EQN = CP03.EQN

AND DMG.SEQN >= CP03.SEQN

GROUP BY CP03.EQN, DMG.SEQN

) RS1,

CP03

WHERE RS1.BLED\_SEQ = CP03.SEQN

GROUP BY RS1.EQN, CP03.BOLN

) IS\_DMG,

( SELECT CP31.EQN, SUBSTR(NVL(CP31.BOLN,CP31.BOOK),1,12) AS BOLN, COUNT(DMGS.SEQN) AS NOF\_PREV\_WDMG

FROM CP03 CP31,

(SELECT EQN, SEQN

FROM CP03

WHERE STA = 'DMG'

) WDMGS

WHERE CP31.STA IN ('LFU','LMT')

AND CP31.MVDT >= 20200101

AND CP31.EQN = WDMGS.EQN(+)

AND CP31.SEQN > WDMGS.SEQN(+)

GROUP BY CP31.EQN, SUBSTR(NVL(CP31.BOLN,CP31.BOOK),1,12)

) PREV\_DMG,

( SELECT CP31.EQN, SUBSTR(NVL(CP31.BOLN, CP31.BOOK),1,12) AS BOLN, COUNT(DFUS.SEQN) AS NOF\_PREV\_DFUL

FROM CP03 CP31,

(SELECT EQN, SEQN

FROM CP03

WHERE STA = 'DFU'

) DFULS

WHERE CP31.STA IN ('LFU','LMT')

AND CP31.MVDT >= 20200101

AND CP31.EQN = DFULS.EQN(+)

AND CP31.SEQN > DFULS.SEQN(+)

GROUP BY CP31.EQN, SUBSTR(NVL(CP31.BOLN,CP31.BOOK),1,12)

) PREV\_DFUL,

( SELECT RS1.EQN, RS1.BOLN,

CASE WHEN CP31.STA = 'PACK' THEN 'PORT'

ELSE 'CUST'

END AS PACK\_AREA

FROM ( SELECT CP31.EQN, SUBSTR(NVL(CP31.BOLN,CP31.BOOK),1,12) AS BOLN, MAX(RELEASE.SEQN) AS RELEASE\_SEQ

FROM CP30 CP31,

( SELECT EQN, SEQN

FROM CP30 CP31

WHERE STA IN ('LSH','SAU','LRE','PCK')

) RELEASE

WHERE CP31.STA IN ('LFU','LMT')

AND CP31.MVDT BETWEEN 20200101 AND 20201231

AND CP31.EQN = RELEASE.EQN(+)

AND CP31.SEQN > RELEASE.SEQN(+)

GROUP BY CP31.EQN, SUBSTR(NVL(CP31.BOLN,CP31.BOOK),1,12)

) RS1,

CP03 CP31

WHERE RS1.EQN = CP31.MQEQN(+)

AND RS1.RELEASE\_SEQ = CP31.MQSEQN(+)

) PACK,

( SELECT RS1.EQN, RS1.BOLN,

CASE WHEN CP31.MQSTA = 'UNPK' THEN 'PORT'

ELSE 'CUST'

END AS UNPACK\_AREA

FROM ( SELECT CP31.EQN, SUBSTR(NVL(CP31.BOLN,CP31.BOOK),1,12) AS BOLN, MIN(RECEIVE.SEQN) AS RECEIVE\_SEQ

FROM CP03 CP31,

( SELECT EQN, SEQN

FROM CP03 CP31

WHERE STA IN ('CCN','NCN','ECP','UPK')

) RECEIVE

WHERE CP31.STA IN ('LFU','LMT')

AND CP31.MVDT BETWEEN 20200101 AND 20201231

AND CP31.EQN = RECEIVE.EQN(+)

AND CP31.SEQN < RECEIVE.SEQN(+)

GROUP BY CP31.EQN, SUBSTR(NVL(CP31.BOLN,CP31.BOOK),1,12)

) RS1,

CP03 CP31

WHERE RS1.EQN = CP31.EQN(+)

AND RS1.RECEIVE\_SEQ = CP31.SEQN(+)

) UNPACK

WHERE MVOY = VOYN

AND MVES = VESS

AND PSEQ = PCSQ

AND STAT = 5

AND SORC = 'CC'

AND BOOKNGTYPE!= 'F'

AND VBLNO = SBLNO

AND EQNO = CP62.EQUIP\_NO(+)

AND BLNO = COMMODITY.BLNO(+)

AND SUBSTR(BLNO,1,12) = IS\_DMG.BOLN(+)

AND EQNO = IS\_DMG.EQN(+)

AND SUBSTR(BLNO,1,12) = PREV\_DMG.BOLN(+)

AND EQNO = PREV\_DMG.EQN(+)

AND SUBSTR(BLNO,1,12) = PREV\_DFU.BOLN(+)

AND EQNO = PREV\_DFU.EQN(+)

AND SUBSTR(BLNO,1,12) = PCK.BOLN(+)

AND EQNO = PCK.EQN(+)

AND SUBSTR(BLNO,1,12) = UNPCK.BOLN(+)

AND EQNO = UNPCK.EQN(+)