



**UNIVERSITY OF PIRAEUS - DEPARTMENT OF INFORMATICS**

ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ – ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

**MSc «Informatics»**

ΠΜΣ «Πληροφορική»

**MSc Thesis**

Μεταπτυχιακή Διατριβή

<b>Thesis Title:</b> Τίτλος Διατριβής:	<b>Customer Behavior Prediction</b> Μοντέλα Πρόβλεψης Συμπεριφοράς Καταναλωτή
<b>Student's name-surname:</b> Όνοματεπώνυμο φοιτητή:	<b>Efthymios Makrandreou</b> Ευθύμιος Μακρανδρέου
<b>Father's name:</b> Πατρώνυμο:	<b>Athanasios</b> Αθανάσιος
<b>Student's ID No:</b> Αριθμός Μητρώου:	ΜΠΠΛ/20045
<b>Supervisor:</b> Επιβλέπων:	<b>Dionisios Sotiropoulos, Assistant Professor</b> Διονύσιος Σωτηρόπουλος, Επίκουρος Καθηγητής

January 2023/ Ιανουάριος 2023

---

**3-Member Examination Committee**

Τριμελής Εξεταστική Επιτροπή

**D. Sotiropoulos**

**Assistant Professor**  
Διονύσιος Σωτηρόπουλος  
Επίκουρος Καθηγητής

**G. Tsihrintzis**

**Professor**  
Γεώργιος Τσιχριντζής  
Καθηγητής

**E. Sakkopoulos**

**Associate Professor**  
Ευάγγελος Σακκόπουλος  
Αναπληρωτής Καθηγητής

## **Acknowledgement**

*First and foremost, I would like to express my sheer gratitude and appreciation for my supervisor and thesis committee chair Mr. Dionisios Sotiropoulos for all the guidance, leadership, patience, motivation, and support with which he has provided all this year.*

*Secondly, I would like to thank The University of Piraeus for offering me all this knowledge during these 2 years learning journey.*

*Last but not the least, I would like to thank my colleague and friends, Eleni, Achilleas, Theseus, Dimitris, Afroditi for everything they have done through these years for me. They supported me a lot in tough times during my research.*

*A big thank you to my family for the encouragement.*

*Once again, I would like to thank everyone without whom this would not be possible.*

*Efthymios (Makis) Makrandreou*

*January 2023*

*Athens*

# Περίληψη

Ζούμε στην εποχή που η επιστήμη της μηχανικής μάθησης ανθίζει καθημερινά για να δώσει λύσεις σε κάθε είδους επιχειρησιακή ανάγκη. Οι ανάλυση δεδομένων έρχεται να δώσει απαντήσεις και κατευθύνσεις σε στρατηγικές που για πολλά χρόνια παίρνονταν εμπειρικά. Δύο είναι οι κύριοι παράγοντες που συμβάλουν στην ραγδαία εξέλιξη του κλάδου της ανάλυσης των επιχειρησιακών δεδομένων, ο μεγάλος όγκος πληροφορίας και η ταχύτητα στην επεξεργασία και την συλλογή. Πολλές είναι οι επιχειρήσεις που πλέον στήνουν στρατηγικές και λειτουργούν βάσει της επιστήμης των δεδομένων. Στην παρούσα διατριβή προσπαθούμε να προσεγγίσουμε την λύση μιας πραγματικής επιχειρησιακής ανάγκης. Πιο αναλυτικά, χρησιμοποιώντας την γλώσσα προγραμματισμού Python, μέσα από μία ακολουθία ενεργειών που έχει πραγματοποιήσει ένας πελάτης, επιχειρούμε με τεχνικές μηχανικής μάθησης και βαθιάς μάθησης να προβλέψουμε την επόμενη ενέργεια του και να προσδιορίσουμε αν ο συγκεκριμένος πελάτης πρόκειται να εγκαταλείψει την υπηρεσία/προϊόν. Σύμφωνα με τα αποτελέσματα της μελέτης, γίνεται φανερό ότι η εκπαίδευση μοντέλων για πρόβλεψη της συμπεριφοράς του πελάτη μέσα από την ακολουθία των προηγούμενων ενεργειών του γύρω από μια υπηρεσία/προϊόν, μπορεί να οδηγήσει σε αποτελεσματικές στρατηγικές πωλήσεων και στοχευμένες προωθητικές ενέργειες. Για την υλοποίηση της εφαρμογής, χρησιμοποιήθηκαν εργαλεία όπως, οι γλώσσες R και το RStudio για την συλλογή δεδομένων και την μετατροπή τους σε μορφή εύκολης και άμεσης επεξεργασίας για τα μοντέλα. Τα μοντέλα είναι στημένα με Python, και πιο συγκεκριμένα έγινε χρήση της βιβλιοθήκης Tensorflow2, Keras και της μεθόδου νευρωνικών δικτύων LSTM. Σαν editor χρησιμοποιήσαμε το Google Collab για λόγους που θα αναπτύξουμε παρακάτω. Τέλος, γίνεται λόγος για την οπτικοποίηση των αποτελεσμάτων με χρήση web app μέσω της βιβλιοθήκης Streamlit της Python.

## Abstract

As the science of Machine Learning evolves, it is a fact that finds the answers to many business needs. The huge amount of data, the fast way of collection and the fast processes of analysis are the main triggers to this. Day by day, a lot of companies already realize the power of data analytics. The subject of this dissertation is the behavioral prediction of a customer around a product/service, taking into consideration the sequence of activities of the customer on it. Briefly, we are trying to predict customer's the next activity and churn alert flag through a sequence of activities using the python programming language. The results show that deep learning techniques can guide companies to new strategies and create targeting campaigns of communication or sale. We used many tools and methods to implement this thesis. First, we used R and RStudio to fetch the necessary first party data and to transform them more easily to the desired structure. The main work is done using python, and more specifically, we focused on Tensorflow2, Keras libraries and we used the LSTM technique. At the end, we mention the importance of visualization and we try to implement a web app to serve the results of the model with an easy and self-service way to the possible stakeholders. As of conclusion, we talk about the output of each experiment and mention some thoughts for future work.

**Keywords**

deep learning, machine learning, customer behavior, prediction, sequence to sequence, python, tensorflow2, neural network, LSTM, keras, churn, classification, model, business case, cx, business intelligence

## *TABLE OF CONTENTS*

Περίληψη .....	4
Abstract .....	4
Keywords .....	5
1. Introduction .....	11
1.1. What is Customer Behavior .....	12
1.2. The importance of customer behavior prediction.....	12
1.2.1. Precise segmentation of audiences .....	13
1.2.2. Personalized customer experiences.....	13
1.2.3. Focused messaging .....	14
1.3. Using Machine Learning to Predict Customers' Behavior .....	15
1.4. Customer Behavior Prediction with RNN - LSTM .....	16
1.5. Computing Platform and Tools .....	17
1.6. Aim .....	18
2. Related Work .....	19
2.1. Smart Home Energy Management System .....	19
2.2. Real time action detection and prediction in human motion streams .....	20
2.3. Predicting users churn on streaming services .....	21
3. Experimental Data .....	23
4. Recurrent Neural Networks.....	27
4.1. Recurrent Neural Network and Deep Learning.....	27
4.2. The basic RNN cell .....	27
4.3. When to use an RNN? .....	29
4.4. RNN topologies .....	30
4.5. Sequence Data .....	32
4.6. The Sequence Learning.....	33
4.6.1. Sequence-To-Sequence .....	33
4.7. Long Short-Term Memory (LSTM).....	34
4.8. Why to use LSTM .....	36
4.9. Why LSTM outperform RNN .....	37
5. Customer Behavior Modeling.....	39
5.1. Experiment 1 .....	39
5.1.1. Data Pre-process .....	39
5.1.2. Tokenization.....	40
5.1.3. Padding .....	40
5.1.4. Model Training .....	41
5.1.5. Working on a Granular Level .....	43
5.2. Experiment 2 .....	45
5.2.1. Data Pre-processing .....	45

5.2.2.	Data Partitioning .....	46
5.2.3.	Min-Max Scaling .....	46
5.2.4.	One Hot Encoding.....	47
5.2.5.	Built the Model .....	47
5.3.	Experiment 3: Customer Churn .....	49
5.3.1.	Aim .....	50
5.3.2.	Data Pre-process .....	50
5.3.3.	Experimental Set-Up.....	50
5.3.4.	Model .....	51
6.	Visualization with Streamlit .....	53
6.1.	Why to serve the results via WebApp.....	53
6.2.	Python Streamlit.....	54
6.3.	Implementation .....	56
7.	Conclusion .....	59
7.1.	Outcome.....	59
7.2.	Future Work .....	60
	Bibliography .....	62

## ***NOMENCLATURE***

AI: Artificial Intelligence.....	5, 4
API: Application Programming Interface .....	5, 14
CPU: Central Processing Unit.....	5
CRM: Customer Relationship Management.....	5, 4, 19, 25
CRO: Conversion Rate Optimization .....	5
CTA: Call to Action.....	5
CTL: Customer Lifetime .....	5, 15, 20, 21
cx: Customer Experience.....	3, 6
DL: Deep Learning .....	5, 20, 23
GDPR: General Data Protection Regulation .....	5, 3, 26
GRU: Graphics Processing Unit .....	5, 9
LSTM: Long short-term memory.....	passim
ML: Machine Learning.....	5, 15, 20, 23
NLP: Natural Language Processing.....	5, 8, 26
RNN: Recurrent Neural Network.....	passim
SMS: Short Messaging Service.....	5, 4, 5
TPU: Tensor Processing Unit.....	5



“Change is the end result of all true learning.”

— Leo Buscaglia



# 1. Introduction

In 2023, the field of Data Analytics and more specific the Customer Behavior Analytics field is a must for each company that wants to thrive and to survive in a world full of competitors (Bazylevska, 2021). During the last 3 years, because of the economic crisis and the Covid-19 expansion, the demand for the customer retention management has become imperative need in every industry (ResearchDive, 2022). The discounts and promo sales campaigns are not the playmaker in strategic plans anymore. The most thriving companies have started to follow a customer-centrism strategy, trying to split into groups their customers with the aim to understand their needs in a better way, to communicate them in a better way and to offer them what they need when they need it (Dago Diedrich, 2021). The above is confirmed by the Forbes' list of 100 customer-centrism companies, as it reveals that many well-known brand names like Apple, Amazon and DHL prove the impact of customer-centrism approach (Morgan, 2022).

Briefly have to say that in this thesis we are focusing on the activities that customers do around a product/service. For ease of understanding and better explanation, from now on, let's assume that we are talking about a company that provides a service as is a subscription. Customers that purchase this service, can permit the company to track their activities (GDPR consents accepted). Some of the customers' activities are the following: extending the service, report a troubleshooting issue, open an email from the company, etc. We will elaborate more about it in the following chapters.

The comprehension of customer behavior is a super-hot topic of discussion not only in companies' meeting rooms (Radu, 2022) but in research too. That is the reason why we decided to make a deep dive around this topic in this dissertation. We focused on an ideal approach trying to satisfy the business need of a company around customer behavior analytics. We created a deep learning model which can predict the next activity of customer mugging on a sequence of activities that the customer made before. We also used the sequence of activities that customers did, to predict if the customer is a churner or not. Both experiments will be completely explained in the following chapters.

For the discussed experiments, we worked with some of the latest techniques and open-source tools with the aim to achieve the best results. From the programming language until the editor and the business thinking we tried to follow the business and tech trends to produce a modern solution (Patel, 2022). In conclusion, the implementation looks more than decent as satisfy this kind of queries either in business world or in academic stage. The real dataset that we used as source, verifies once more the legitimacy of our way of working.

## **1.1. What is Customer Behavior**

As its core, customer behavior (Jane Priest, 2013) is the study of how people make buying decisions. It attempts to understand how they choose, use, and dispose products/services, as well as the various stages people go through before making a purchase or an interaction around a good or service. As mentioned before there are several factors that affect customers' decisions (cultural, social, economic, geographical, psychological) (Sengupta, 2022). The study of customer behavior is a hot trend nowadays as the companies give a daily battle in competition area. Customer Behavior analysis can be implemented by various ways as is the basket analysis, the clustering techniques, churn prediction and many more others (Fontanella, 2022).

You probably know your best friends well — their likes, dislikes, where they shop, why they prefer certain brands, and even what they would buy before walking into a store. The companies of now, want to be their customers' best friend. Walking in the customer's shoes helps identify touchpoints to collect data from. Every customer interaction has a rich trail of data, from product choices to channel preferences to when and why they want to buy from you, versus the competition. Valuable sources of data include (but are not limited to) mail clicks, in-store visits, online purchases or browsing, and content streaming. In this thesis we worked using some of the customers' digital footprints. Meaning that we used the sequence of activities that the customer did around the goods/services.

## **1.2. The importance of customer behavior prediction**

If a brand needs to survive and thrive in this ultra-competitive era, it needs to understand social proof and behavioral influences to have a good grip around its customers. Customer behavior prediction utilizes the numerous "digital (online) and offline (not online) footprints" left by customers to help brands stay a step ahead of them (Farshad Kooti, 2016). The more informed you are, the smarter your decisions will be. Data about our browsing and buying patterns are everywhere. From credit card transactions and online shopping cards to customer loyalty programs and user-generated ratings/reviews, there is a staggering amount of data that can be used to describe our past buying behaviors, predict future ones, and prescribe new ways to influence future purchasing decisions. Diving deep into customer analytics does not just benefit the business by helping it make data-based decisions to increase sales and drive product growth—it benefits the customers, too, as it helps to learn how to better meet user needs (Grace, 2021). A strong customer analytics process helps to understand who the customers are, how they behave, and how satisfied they are with the company or product experience. Using customer analytics helps to discover which touchpoints in the customer experience are most important for personalized marketing and how to implement these specialized interactions across the customer journey. Creating personalized customer content can be tricky, so it is important to have spot-on data that will effectively drive conversions, WOM(word of mouth) and engagement. Through predictive analysis drawn from data of interaction and transaction behaviors, customer analytics can help reduce customer churn and increase loyalty. This may sound like a bunch of technical talk but all it means is that studying what your customers have done in the past will help you better serve them in the future (Rohn, 2022).

Below are the three main benefits of predictive customer behavior:

### 1.2.1. Precise segmentation of audiences

While CRM systems provide basic information about customers, predictive analysis pulls dynamic data about a customer's evolving tastes and behaviors (Murariu, n.d.). Numerous variables are weighed to create a fluid customer segments. This segment is more meaningful than an obsolete one, which is based on vague demographics and past transactions. Equipped with the power to predict, businesses can define the actual "value" of a customer, and then make an efficient decision on whether it makes sense to invest effort and time to nurture a customer too. In this case, brands can identify high-value groups that are amenable for further selling tactics, such as up-selling and cross-selling.

Intelligent segmentation/clustering can also direct brands towards buyer groups that are more willing to share their positive shopping experiences with others. With little nurturing, these spenders can become brand promoters and advocates. They can help create a positive brand image and bring in new customers – overall resulting in more success in revenue (Yasar, 2021). Here are the four primary categories of customer segmentation, examples listed as restaurant guest segments (Musumano, 2019):

<b>Geographic</b>	<b>Demographic</b>	<b>Behavioral</b>	<b>Psychological</b>
Based on a customer's home location	Based on age, income, occupation, family size	Based on a customer's purchasing habits	Based on a customer's beliefs and values
<b>EXAMPLE</b>	<b>EXAMPLE</b>	<b>EXAMPLE</b>	<b>EXAMPLE</b>
<i>Locals vs. tourists</i>	<i>Families vs. couples vs. business diners</i>	<i>High-Spenders vs. Low-Spenders</i>	<i>"Here for Instagram" vs. "Here for the freebies"</i>

Figure 1: segmentation is the process of dividing a broad consumer or business market, normally consisting of existing and potential customers, into sub-groups of consumers based on some type of shared characteristics.

Accurate customer segmentation allows companies to engage with each customer in the most effective way. For all the above, customer segmentation is a win-win road to success for companies and consumers.

### 1.2.2. Personalized customer experiences

Not all customers have the same expectations from a brand. If a business understands its customers and the factors impacting their behaviors, they can create customer experiences that are designed to delight. Satisfied customers often perform repeat purchases, so this is a sure-shot way to secure customer loyalty and retention. Before this topic, brands relied on guesswork or assumptions to gather customer intelligence. Then they shifted to vague data sources such as point-of-sale or customer demographics. Today, brands are using advanced channels such as AI, device-generated data, alongside behavioral detection tests driven by the customers themselves, like AB Testing strategies, to understand their audience better (Starikova, 2018). Earlier, brands relied on guesswork or assumptions to gather customer intelligence. Then they shifted to vague data sources such as point-of-sale or

customer demographics. Customer behavior prediction rests on deep learning, which is a subset of AI. Deep learning involves building a layered (or neural) algorithm that can process mammoth databases of variables. When marketers feed variables about past, existing, and potential customers into the deep learning model, they get a near-perfect prediction about the tastes and motives of each customer. Armed with this AI-powered knowledge, brands can come up with content, products, and other offerings that are tailored to meet the expectations of each target group and convert them faster. Platforms like Cortex, Hootsuite, and Zoho Social have AI-enabled features that can optimize your content according to audience needs and preferences (Shirk, 2021). For example, Cortex analyzes industry trends and keeps you updated with your industry's visual language. It also discovers visual themes, colors, features, and composition that works best for your audience. To illustrate this point better, let us consider the example of Netflix. Netflix claims that its algorithm is so strong that it influences 80% of the total content consumed by its subscribers and saves more than one billion USD yearly in value from customer retention (Chong, 2020).

### **1.2.3. Focused messaging**

A spray-and-pray messaging approach does not work anymore. Sending bulk SMS and emails to the entire contact list just produces high costs with little returns. A brand that keeps a close watch on their customers' buying patterns gets a fair idea about their next steps and can send event-triggered messages (Williams, 2019).

These automated text messages require minimal effort, and when utilized well, you can see a clear boost in conversions using them – abandoned cart text messages fall under this bracket of SMS re-marketing. For instance, let's say your website has numerous regular visitors who love browsing through your product catalogs but don't buy anything. Through customer behavior prediction, you can identify this visitor subset and know exactly at which point they will drop off the sales funnel. You can accordingly make the necessary changes required to get them to become paying customers.

Brands can draw maximum mileage from customer insights by refreshing your SMS marketing with email marketing (Osborne, 2021). Following up a triggered SMS with a series of well-timed and compelling emails can bring down cart abandonment rates and re-engage lost website visitors. Find out when is the best time to send abandoned cart email here. Studying customer behavior indicates the most active times of audiences and the email subject lines that have the highest success rate. Marketers can use this information to optimize their email marketing strategy. Conversion rate optimization (or CRO services) is another effective tactic that you can use to expedite conversions through focused messaging. Limited-time voucher codes and countdown timers help create a sense of urgency, otherwise known as call to action or e-commerce CTA, for your website visitors and speed up their conversion. Here again, predictive analysis of a visitor's response can help you offer the most engaging deals. So, imagine receiving a personalized communication from a company that satisfies your needs the right moment. Could you be a promoter for this company? Sure, you could! In Figure 2, you may find a realistic custom conversation between two friends and customers that received different communication offers/messages from two companies.

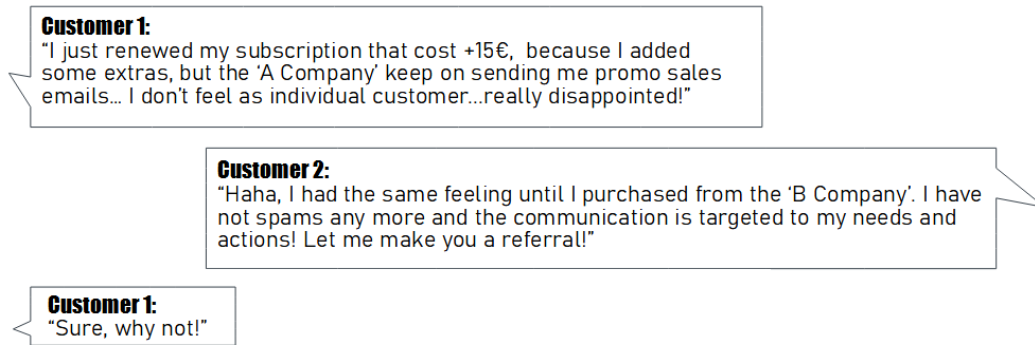


Figure 2: a custom conversion between two different types of customers, a happy one and dissatisfied

### 1.3. Using Machine Learning to Predict Customers' Behavior

Undoubtedly, it is a fact that the most of the business companies' decisions are influenced (or should be) by the results derived from studying the customer behavioral data of online or offline customers by experts in data analysis, data science, and machine learning. Suppose the managerial team of an online retail shop approaches you, a data scientist, with the dataset wanting to know whether customers will make their next activity from the day they made their last purchase (Pilladin, 2021). Your answer to their inquiry will help them identify which customers their marketing team need to have a focus on regarding the next promotional offers they will be rolling out. Taking decisions and making plans are like betting. Having data is your running start.

In this thesis, my goal is to build a model or more that will provide a suitable answer to the question posed by the firm's managers. More precisely, using the given dataset, I build a deep learning model that predicts the next action of a customer and flags customers as potential churners or active users regarding their sequence of past activities. For better comprehension, please see the bellow custom figure, which presents a sequence of 5 activities, described in detailed. The previous 5 activities could drive into a specific 6<sup>th</sup> (last) activity. The goal is to predict the last activity with the highest possible accuracy.

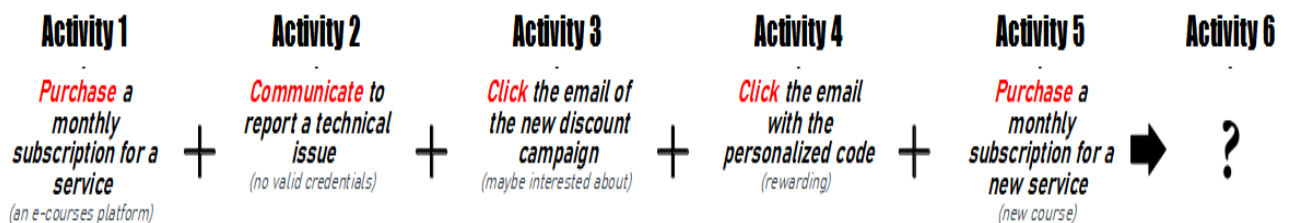


Figure 3: custom of customer sequence of activities

## 1.4. Customer Behavior Prediction with RNN - LSTM

Till now we focused on generic customer behavior prediction with Machine Learning techniques. Following the actionable part where we will try to deep dive more.

Predicting the next activity is a neural application that uses Recurrent neural networks. Time series prediction problems are a difficult type of predictive modeling problem. Unlike regression predictive modeling, time series also add the complexity of a sequence dependence among the input variables. Since basic recurrent neural networks have a lot of flows, we go with LSTM. A powerful type of neural network designed to handle sequence dependence is called a recurrent neural network. Because very large architectures may be effectively taught, the Long Short-Term Memory network, often known as the LSTM network, is a type of recurrent neural network used in deep learning. Here, with the aid of those three gates we previously observed, we may ensure that we have a longer recollection of what tasks are vital (Brownlee J. , 2016). To get as close as we can to the "ideal" solution, we will work with three distinct versions and presumptions (3 experiments).

A briefly mathematic but simple explanation for the sequence 2 sequence experiments could be the above. In the event sequence prediction task, the dataset is a set of sequences  $S = \{s_1, s_2, \dots, s_n\}$ . For  $i$ -th sequence  $s_i = h(e^{(i)}_1, a^{(i)}_1, a^{(i)}_2, \dots, a^{(i)}_{m(i)}), (e^{(i)}_2, a^{(i)}_2, a^{(i)}_2, \dots, a^{(i)}_{m(i)}), \dots, i$ . Here,  $e^{(i)}_t$  represents the  $t$ -th event in sequence  $s_i$ .  $a^{(i)}_k$  represents the  $k$ -th attribute of event  $e^{(i)}_t$ . Depending on specific datasets, the number of attributes varies, the meanings of events and attributes are also different. The problem of predicting next event and its attributes given a sequence of past events and the problem of suffix generation is what we are trying to solve here using a real-world dataset.

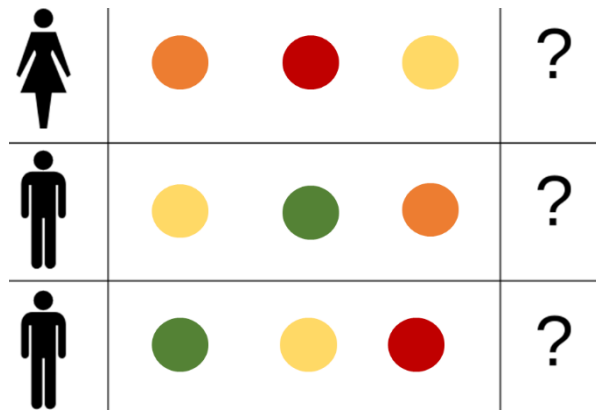


Figure 4: different customers, different sequences, different behaviors.

The above image presents the activity types for three customers. All customers made 3 activities. Those activities draw a pattern that drives into the “?” last-upcoming activity. The aim is to define the upcoming activity for each customer. The ‘why’ has been explained before, the ‘how’ is being explained below.



## 1.5. Computing Platform and Tools

It is not fair to skip the computing platform and the tools that we used. Here we will explain what we used and the reason behind the choice of using it.

Google Colab was used as the basic development environment for these experiments. Google Colab provides hosted runtimes for Python development with free access to GPUs and TPUs (Chng, 2022). This was an important feature. Text data can be very resource demanding and the process could not be executed on available local machines. The first try of implementation was on Anaconda suite and more specifically in Jupyter Notebook. The model training was a very slow procedure on that.

If you ever have wanted an easy-to-configure interactive environment to run your machine learning code that came with access to GPUs for free? Google Colab is the answer. It is a convenient and easy-to-use way to run Jupyter notebooks on the cloud, and their free version comes with some limited access to GPUs as well. If you are familiar with Jupyter notebooks, learning Colab will be a piece of cake, and we can even import Jupyter notebooks to be run on Google Colab. But there are a lot of nifty things that Colab can do as well, as speed up, training using Google Colab's free tier with GPU, using Google Colab's extensions to save to Google Drive, present interactive display for pandas DataFrame, etc and save your model's progress when training.

The CNN model was developed with Keras (Simplilearn, 2022), an open-source library built on top of TensorFlow (TensorFlow, 2022) 2 designed to provide a high-level API towards TensorFlow to simplify and speed-up the development of deep learning models. Tensorflow is an end-to-end open-source platform for machine learning. It is a symbolic math library that uses dataflow and differentiable programming to perform various tasks focused on training and inference of deep neural networks. It is worth noting again that the biggest part of pre work on the first party data was executed with R in R studio. So, it is not an exaggeration to say that this thesis bear on a lot of programming tools.

One more critical point to add is that we worked only with open sources tools. Open-source technologies provide more flexibility and less dependence on proprietary platforms, as well as cost savings. With open source, developers can inspect and modify the source code to fit their needs and requirements. When looking at what is available, there are two main categories:

- Open-source platforms that build a complete cloud environment.
- Open-source tools that manage or administer cloud services running within a proprietary public cloud.

Similar to proprietary services, open-source cloud platforms and tools can help IT teams deploy, provision and manage workloads and environments. However, these offerings typically provide deployment and management options that extend beyond the usual menu of proprietary services, as well as offer a wider range of options for implementing and administering the cloud environment. And because some open-source cloud tools are free - at least in their core, open-source form - enterprises can save money (Tozzi, 2021).

## **1.6. Aim**

Organizations have realized the importance of technology to surpass their annual goals. Moreover, they have already begun integrating concepts like AI and machine learning in their ecosystems. Of all the domains, customer experience is the first one to undergo such a change at a significant level. From customer support to personalized recommendations, AI and machine learning are conquering this domain and are on the verge of complete automation (Sindle, n.d.).

Our goal is to achieve to identify the customer behavior. We aim to a 360 solution, trying to answer queries as the below:

“What could be the next customer activity?”

“Which is the pattern of activities sequence that drives into a vulnerability status?”

We will build a neural model to predict this. The end goal of the application is by importing a sequence of customer activities to predict the upcoming final activity. To be honest, in the conclusion while ML tools are powerful, they still need humans to oversee them to ensure that they stay within bounds, are inline with brand values and identity, and that all teams affected by the algorithm are prepared for the impact.

## 2.Related Work

The problem of learning to predict a customer's next activity is related to work in several areas. There are many similarities in the problems studied in plan and goal recognition in the user modeling community. Event sequence prediction has wide applications on economics, electronic, health, and social media monitoring. Accurate prediction of event sequences can help provide better service to customers and prevent risks. Making research you can easily find out that the LSTM neural (Keith, 2021) network is main way of working on this kind of experiments.

### 2.1. Smart Home Energy Management System

The team of M. Goutham, S. Stockar, R. Blaser and P.D. Hanumalagutti from the University of Columbia, Columbus worked on the Smart Homes using Multi-Layer Long Short-Term Memory Networks study about the in-house next move prediction for each occupant at the background of Smart homes technology trend.

Through the implementation of demand response programs, smart homes offer a special potential for utility companies and end users to increase grid efficiency, dependability, and end-user cost. If a Smart Home Energy Management System is in place and can plan programmed electric loads to take advantage of cheaper electricity costs, utilities can control their customers' electricity demand by offering variable electricity prices. The Smart Home Energy Management algorithm might incorporate a prediction of the activity sequence to increase customer acceptability and close the optimality gap. This research offers a multi-layer Long Short-Term Memory network-based user activity prediction model inside this framework. Using dynamic time warping, the prediction method's accuracy is assessed (M. Goutham, 2021).

The routine of a smart home can be learned using a method that has been devised, and this way can be included into a predictive smart home energy management algorithm to better schedule appliance activation. To learn patterns in user behavior, a classification network with stacked LSTM layers combines time-based activity predictions with transition probability-based activity predictions. The technique may be quickly put into practice using pre-made neural network modules, and it can be changed to consider other pre-processed inputs like frequency of occurrence and last known occurrence for a particular appliance of interest. The efficiency of a prediction for a time horizon can be measured by comparing time-series data using a technique called dynamic time warping. This can be utilized to identify situations where valuable activity predictions are made but offset from the actual time of occurrence by combining it with the conventional accuracy percentage-based selection of neural network parameters. For the dataset under study, a pointwise prediction accuracy of about 60% is reached after a 24-hour forecast period. While this is helpful in ensuring that the time-horizon in the MPC used by the HEM algorithm is considered appropriately, a family with a more closely followed routine would produce higher forecast accuracy. The forecast also depends on the identified activities, which in our case study only depended on the amount of electricity used by various appliances. Greater activity recognition can be achieved with various data kinds, such as proximity and door sensors, which will improve prediction accuracy (M. Goutham, 2021).

The results of the above experiments declare the level of complexity that this kind of problem has. Even the final accuracy score, ~60% foreshadows that our sequence-to-sequence model will not reach the highest level of accuracy.

## **2.2. Real time action detection and prediction in human motion streams**

Another interesting application based on real time action detection and prediction in human motion streams. Fabio Carrara, Petr Elias, Jan Sedmidubsky, Pavel Zezula worked with LSTM on it in 2019.

Digital representations of human movements using motion capture data are made using 3D skeleton configurations in sequence. Such spatiotemporal data, which are frequently stream-based in nature, must be processed effectively in order to identify behaviors of high interest, such as hand gestures in real-time human-computer interaction. The alternative is to persistently retain automatically annotated segments of a continuous stream so that they can be searched for later retrieval or pattern mining. Fabio Carrara, Petr Elias, Jan Sedmidubsky, and Pavel Zezula's research focuses on multi-label recognition of user-specified actions in continuous streams as well as unsegmented sequences. To efficiently encode the skeleton frames within the hidden network states, we specifically make use of recent developments in recurrent neural networks and employ a unidirectional LSTM model. During the training phase, the model discovers which subsequences of the encoded frames fall under the designated action classes. The annotation step then makes use of the previously acquired representations of the classes to estimate the likelihood that an incoming skeleton frame corresponds to a certain action class. Ultimately, a learnt threshold is used to compare the computed probabilities in order to automatically identify the start and finish of actions. Fabio Carrara, Petr Elias, Jan Sedmidubsky, and Pavel Zezula use a bidirectional LSTM model to estimate class probabilities by taking into account both the past and the future frames in order to further improve the annotation accuracy. On the three use cases of real-time stream annotation, offline annotation of lengthy sequences, and early action detection and prediction, they thoroughly analyze both models. The results show that the models are at least one order of magnitude more effective and efficient than the state of the art, being able to annotate 10k frames per second (Fabio Carrara, 2019).

Here, the methodology was to specify the motion capture data and the multi-label annotation issue. In order to accomplish online annotation of streams and offline annotation of sequences, they then introduce two methods (Online-LSTM and Offline-LSTM) based on recurrent neural networks. The team effectively applies both unidirectional and bidirectional LSTM architectures to the challenging challenge of multi-label action recognition in unsegmented skeletal sequences<sup>1</sup>. They were inspired by recent developments in recurrent neural networks. They specifically introduce the Online-LSTM model, which can be used to detect time-critical actions early in skeletal streams, and the Offline-LSTM model, which improves accuracy even more when annotating lengthy sequences offline. The accuracy is assessed using both the threshold-independent average precision (AP), which more accurately captures the relationship between precision and recall across all feasible threshold settings, and the best performing threshold selection (F1 score), as presented in the

majority of related work. The ability to detect actions with a minimum or even no delay is demonstrated only with a slight decrease in precision. Moreover, the ability to predict actions few hundred milliseconds before they happen is observed for annotations having a high IoU with the ground truth (Fabio Carrara, 2019).

It is a subtle but challenging extension of sequence prediction where rather than predicting a single next value in the sequence, a new sequence is predicted that may or may not have the same characteristics or be of the same time as the input sequence. As we see from the above applications, the usage of LSTM satisfies the requirements for variety of AI experiments. Everything could be studied as a sequence-to-sequence prediction exercise. The sequences are everywhere, we just have to make the correct interpretation for each case. Simple? Not!

### **2.3. Predicting users churn on streaming services**

This study has been defined as a master thesis work at Spotify from Helder Marting in KTH Royal Institute of Technology School of Computer Science and Communication in 2017.

The main focus of this thesis was forecasting customer churn, or whether a client will leave a service provider. For comparison purposes, well-known classifiers like logistic regression and random forest were employed as a baseline against a more recent strategy known as the LSTM recurrent neural network. It has been established that, across all tests and models considered, LSTM has the best F1-score metric overall. In comparison to LSTM, random forests have a little advantage in the precision-recall AUC. Although there is a small discrepancy in the analyzed metrics between the two models, random forests and LSTMs are determined to be equivalent churn prediction methods. However, logistic regression performed poorly across all experiments, making it an inappropriate technique for churn identification. The research investigated how different aspects of the training data affected how accurate the predictive models were in the end. The time window used to determine whether the user retained the information was confined to 7 days in the future, with the size of this time window steadily reducing to achieve the maximum harmonic mean between precision and recall for the churning class. When the distribution of the training data over the retained and churned classes was closest to the actual distribution of the test set, at the expense of a poor recall, the maximum value was reached. This ratio between the class labels also had a substantial impact on the F1-score measure. A balanced training dataset is advised if service providers are more concerned with catching the bulk of prospective churners. The architecture for building a dataset appropriate for training churn prediction models was presented at the end, along with a basic analysis of its features that provides a broad overview of the user cohort of the service provider's behavior. This could be further explored by the company as to develop retention actions that could reduce levels of churn and thereby increasing profit.

So, in conclusion, the LSTM neural network seems to be the best approach to work on customer behavior prediction.

“The LSTM cell adds long-term memory in an even more performant way because it allows even more parameters to be learned. This makes it the most powerful [Recurrent Neural Network] to do forecasting, especially when you have a longer-term trend in your data. LSTMs are one of the state-of-the-art models for forecasting at the moment” (Kostanje, 2021)

The above experiments are all LSTM applications of Deep Learning with very interesting content and purpose. Each one has different focus, features and architecture. Our work will be under the perspective of LSTM technique, having a way to stand out because of the data processing, the real dataset usage, the business needs and the 360 approach that we are going to follow.

### 3. Experimental Data

Moving to actionable part, the comprehension of the dataset is matter of importance to easily follow each experiment procedure. The dataset we used is a real one, that came from various data marts of a large-scale company. Briefly, the dataset contains the activities that customer made in a specific time window. In more detail, it is structured by Customer\_ID by date of activity happened by activity type. In this chapter we will elaborate more into dataset as the data is the crucial element for each machine learning application. The below table presents the main structure of the dataset, that has been described earlier. You may see the type of activities happened on each day (sorted by oldest to newest) for an individual Customer (Customer\_ID).

Customer_ID	Date	Activity Type
AZ109001	7/12/2020	E
AZ109001	30/1/2021	A
AZ109001	28/4/2021	B
AZ109001	18/5/2021	A
AZ109001	18/6/2021	C
AZ109001	7/12/2021	B
AZ109001	2/2/2022	C
AZ109001	24/5/2022	A
AZ109001	24/5/2022	B

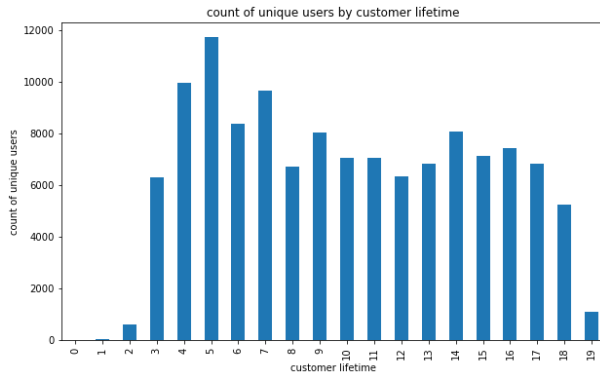
Table 1: Main Dataset structure

In order to fetch the necessary first party data, we connected with 5 different databases. The process to collect the data under the perspective of customer was not so simple. Finally, with the usage of R and RStudio we managed to achieve this unification and create the main dataset, which we will explain below. The volume of customers, the number of activities, the extra columns are meaningful information for the best comprehension of the solution. Plentiful data could provide accurate suggestions (Tkachova, n.d.).

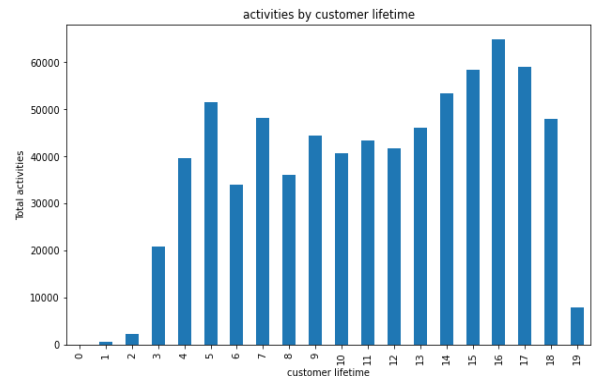
Data collection, data cleansing, join traction and attribute data are some of the steps we followed. Last but not least, the ending task was to create new features by existing one. That is how the Activity Type1 column has been created. Some values in your data set can be complex and decomposing them into multiple parts will help in capturing more specific relationships. This process is actually the opposite to reducing data as you have to add new attributes based on the existing ones. This unlocked the way to go up to the solution.

The main dataset contains in total ~1mio activities which are made by ~150K customers. The activity types have two layers (Type1 and Type2). Type1 contains 8 kinds of activities while Type2 contains 21 kinds. Type2 is a way to give a better explanation in Type1 (ex. The Customer\_ID = 3454BA, in Date=20221011, made an activity Type1=C and in more detail Type2 = C2). The dataset includes customers that interacted with company between '1/1/2021' and '30/05/2022'. Our dataset covers

almost 1-and-a-half-year data. It is obvious that the ‘activity date happened’ time window is the same with the above. For further information and better comprehension of data please see below the number of unique customers grouped by the Customer Lifetime group that they belong.



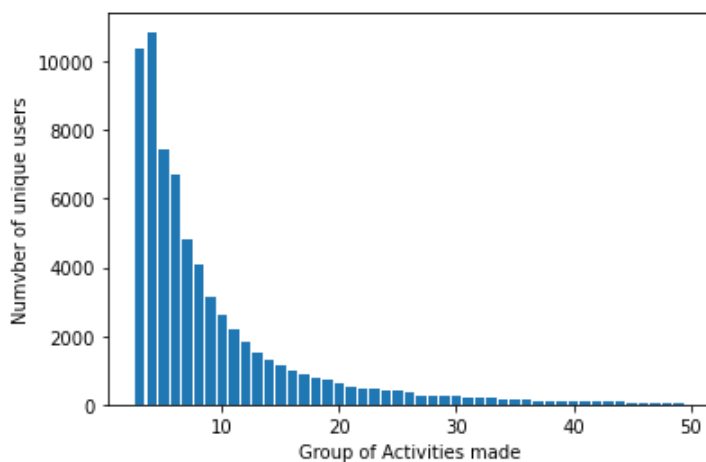
plot 2: this plot presents the number of unique customers by CLT group



plot 1: this plot presents the volume of activities made by users having the relevant CLT

As you can see with the help of the above plots, in our dataset we have legit representation within all customer lifetime group (excluded the 0,1 ,2 and 19 months). This is very important as it gives us the opportunity to create sub datasets and smaller groups of customers to focus on targeted audience and achieve better results and accuracy. It is also interested to see the number of activities made by customer lifetime group, too. From the second plot, we could realize that there is a trend of making more activities around the service as long as the consumer lifetime increased.

One extra point is that there are customers that made 1 activity during the time window and customers that made more than 100. To give a view of it, below you may find the graph regarding the number of customers that made a specific number of activities. The main volume of customers ~120K made up to 25 activities during the discussed period. Through the dataset analysis, there are more than 40K customers with only one activity (excluded from the down plot).



plot 3: this plot shows the volume of unique customers by the aagregation of activitv group



For the application requirements on this thesis, the main (above) dataset is about to be cut as need to target on specific Customer Lifetime Groups of Customers to gain the best results that we can and to work on a more realistic approach.

This is the detailed format of our dataset:

Customer_ID	Customer_LifeTime (months)	Gender	Age_Group	Date (Activity)	Type1	Type2	Churner
3454BA	8	F	18-24	20210813	A	A4	0
3454BA	8	F	18-24	20210919	A	A2	0
3454BA	8	F	18-24	20220204	C	C2	0
6677ZZ	2	M	25-35	20220328	B	B4	0
8003ZQ	12	M	55+	20210603	B	B2	1
8003ZQ	12	M	55+	20210603	E	E2	1
8003ZQ	12	M	55+	20210628	A	A2	1
8003ZQ	12	M	55+	20220103	A	A2	1

Table 2: Dataset preview

The dataset that used in each application had crucial effect to the results. From scratch, the most import step was to define the right dataset. While we are living in very dynamic ages, the companies and their products/services used to be changed and transformed.

So, customers have more and more options around a product. It is a fact that this new reality creates new needs to customers. And as a result, those “new needs” create new habits and demands. It was a bit tricky to keep a specific amount of data (as much as possible) for the period that the service was stable in transformations (not many upgrades, not many new options) with aim to focus on equivalent customers behavior patterns.

So, let’s make a revision to avoid any misclassifications and to keep the important points of the above. The main dataset is a combination of 5 systems data. To fetch and transform the data into a comfortable structure for the analysis we used R and RStudio. The rest work is done with python, in google collab. The dataset except from being updated (2021-2022 data) is also sufficient to satisfy the business needs and technical requirements for a successful ML/AI application. 150.000 customers made more than 1mio activities (rows) allow us to have the ability to work without limitation and restrictions. Anywise, working on a ML application is trial & error process.

The fact that the data used are real gives the chance of approach the solution with a good comprehension. Using a real-world dataset for the project offers the following benefits. Real data comes with its own challenges hence good opportunity to learn about the typical data issues and about handling them. Real dataset is easier to understand and relate to the dataset. Dynamics in the attribute features could drive further learning. Helps to differentiate your work and hence your profile as well. Of course, real dataset is different from toy examples. The first challenge that I faced was to modify the dataset correctly, data pre-processing took about ~40%-50% of the whole solution time. Hanging with real data, I had to pay attention to the timestamps, the chronology of path is extremely important. Customers are more interested in which paths lead to conversion rather than which has not. Hence it makes sense to break

your data set into two (e.g., churners vs no churners). At the beginning, I tried to experiment the whole dataset. So, I realize that I have to apply the algorithm selectively. Applying the algorithm to the whole data set (sometimes having millions of records) lead to intractable solution. The removal effect is a step that I had to take next. Data and hanging data are a very essential thing. In the end, achieving excellency in data quality is an art of its own, and what it is covered above it is just the tip of the iceberg.

## 4. Recurrent Neural Networks

In this chapter, we will focus on Recurrent Neural Networks (RNNs), a class of neural networks that are popularly used on text inputs. RNNs are very flexible and have been used to solve problems such as speech recognition, language modeling, machine translation, sentiment analysis, and image captioning, to name a few. RNNs exploit the sequential nature of their input. Text, speech, time series, and other types of inputs that depend on the pieces that came before them in a sequence are all examples of sequential inputs. First, we will examine a basic RNN cell's internals to see how it handles the input's sequential dependencies. Additionally, we will discuss several drawbacks of the fundamental RNN cell (implemented as SimpleRNN in Keras) and examine how the Long Short-Term Memory (LSTM) in SimpleRNN solves these drawbacks (Gulli, 2019).

### 4.1. Recurrent Neural Network and Deep Learning

Neural network is about logistic regression. As a matter of fact, the logistic regression model, or rather its generalization for multiclass classification, called the SoftMax regression model, is a standard unit in a neural network. Recurrent neural networks (RNNs) are the state-of-the-art algorithm for sequential. It is the first algorithm that remembers its input, due to an internal memory, which makes it perfectly suited for machine learning problems that involve sequential data. It is one of the algorithms that helped deep learning accomplish some incredible successes over the past several years. The fundamental principles of how recurrent neural networks operate, as well as the main problems they face and their solutions, will all be covered in this thesis.

RNNs are a powerful and robust type of neural network and belong to the most promising algorithms in use because it is the only one with an internal memory.

Recurrent neural networks are a common deep learning approach that has been around for a while. Although they were initially developed in the 1980s, it was not until recently that we truly realized their potential. RNNs have truly come to the forefront thanks to improvements in computing power, the enormous amounts of data we now have to deal with, and the development of long short-term memory (LSTM) in the 1990s.

Because of their internal memory, RNNs can remember important things about the input they received, which allows them to be very precise in predicting what is coming next. Therefore, they are the preferred algorithm for sequential data like time series, speech, text, financial data, audio, video, weather and much more. Recurrent neural networks can form a much deeper understanding of a sequence and its context compared to other algorithms.

### 4.2. The basic RNN cell

The basic RNN cell Traditional multilayer perceptron neural networks assume that all inputs are independent of each other. This assumption is not true for many types of sequence data. For example, words in a sentence, musical notes in a composition, stock prices over time, or even molecules in a compound, are examples of sequences where an element will display a dependence on previous elements (Gulli, 2019).

RNN cells incorporate this dependence by having a hidden state, or memory, that holds the essence of what has been seen so far. The value of the hidden state at any point in time is a function of the value of the hidden state at the previous time step, and the value of the input at the current time step, that is:

$$h_t = \phi(h_{t-1}, X_t)$$

Here,  $h_t$  and  $h_{t-1}$  are the values of the hidden states at the time  $t$  and  $t-1$  respectively, and  $x_t$  is the value of the input at time  $t$ . Notice that the equation is recursive, that is,  $h_{t-1}$  can be represented in terms of  $h_{t-2}$  and  $x_{t-1}$ , and so on, until the beginning of the sequence. This is how RNNs encode and incorporate information from arbitrarily long sequences. We can also represent the RNN cell graphically as shown in Figure below. At time  $t$ , the cell has an input  $x(t)$  and output  $y(t)$ . Part of the output  $y(t)$  (represented by the hidden state  $h_t$ ) is fed back into the cell for use at a later time step  $t+1$ .

Just as in a traditional neural network, where the learned parameters are stored as weight matrices, the RNN's parameters are defined by the three weight matrices  $U$ ,  $V$ , and  $W$ , corresponding to the weights of the input, output, and hidden states, respectively. Recurrent neural networks can model sequential information. They do not assume that the data points are intensive. They perform the same task from the output of the previous data of a series of sequence data. This can also be thought of as memory. RNN cannot remember from longer sequences or time. It is unfolded during the training process, as shown in the following figure 5 (Shanmugamani, 2018):

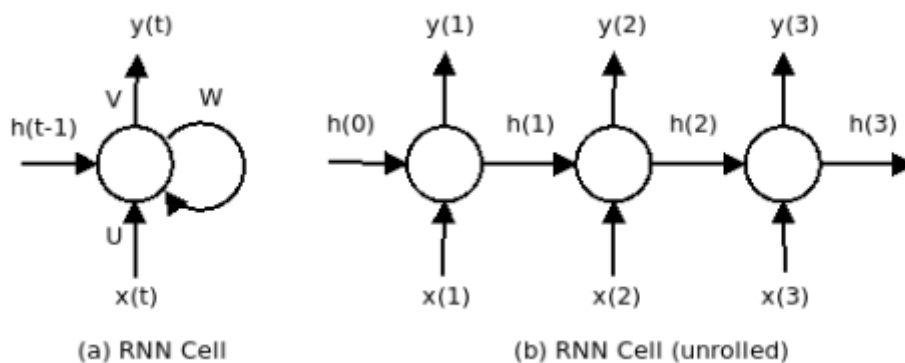


Figure 5: Schematic of an RNN cell and RNN cell unrolled

Figure 5(b) shows the same RNN in an "unrolled view". Unrolling just means that we draw the network out for the complete sequence. The network shown here has three-time steps, suitable for processing three element sequences. Note that the weight matrices  $U$ ,  $V$ , and  $W$ , that we spoke about earlier, are shared between each of the time steps. This is because we are applying the same operation to different inputs at each time step. Being able to share these weights across all the time steps greatly reduces the number of parameters that the RNN needs to learn (Gulli, 2019).

We can also describe the RNN as a computation graph in terms of equations. The internal state of the RNN at a time  $t$  is given by the value of the hidden vector  $h(t)$ , which is the sum of the weight matrix  $W$  and the hidden state  $h_{t-1}$  at time  $t-1$ , and the product of the weight matrix  $U$  and the input  $x_t$  at time  $t$ , passed through a tanh

activation function. The choice of tanh over other activation functions such as sigmoid has to do with it being more efficient for learning in practice, and helps combat the vanishing gradient problem, which we will learn about later in the chapter.

### 4.3. When to use an RNN?

“Whenever there is a sequence of data and that temporal dynamics that connects the data is more important than the spatial content of each individual frame.”

– Lex Fridman (MIT)

You must have a basic understanding of sequential data and "regular" feed-forward neural networks in order to comprehend RNNs effectively. Fundamentally, sequential data is essentially ordered data in which related items are placed one after the other. The DNA sequence or financial data are two examples. Time series data, which is just a collection of data points that are listed in chronological order, may be the most well-known sort of sequential data. Feedforward networks were the first type of artificial neural network devised. In this network the information moves in only one direction, forward. Input nodes receive data and pass it along, as seen in Figure 6 (Horowitz, 2014).

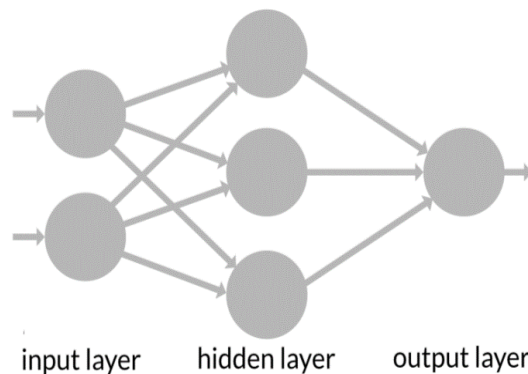


Figure 6: RNN nodes - flows

RNNs and feed-forward neural networks get their names from the way they channel information. In a feed-forward neural network, the information only moves in one direction — from the input layer, through the hidden layers, to the output layer. The information moves straight through the network. Feed-forward neural networks have no memory of the input they receive and are bad at predicting what is coming next. Because a feed-forward network only considers the current input, it has no notion of order in time. It simply cannot remember anything about what happened in the past except its training. In an RNN the information cycles through a loop. When it plans, it considers the current input and what it has learned from the inputs it received previously. The two images below, figure 7, illustrate the difference in information flow between a RNN and a feed-forward neural network (Choubey, 2020).

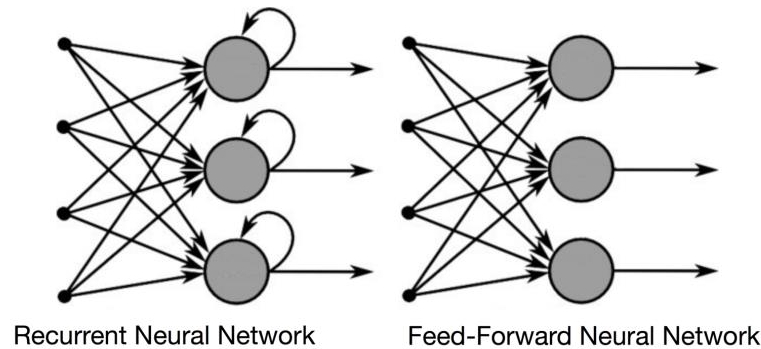


Figure 7: difference between an RNN and a feed-forward neural network.

A usual RNN has a short-term memory. In combination with a LSTM, they also have a long-term memory (more on that later). Another good way to illustrate the concept of a recurrent neural network's memory is to explain it with an example: Imagine you have a normal feed-forward neural network and give it the word "informatics" as an input, and it processes the word character by character. By the time it reaches the character "o," it has already forgotten about "i," "n" and "f," which makes it almost impossible for this type of neural network to predict which character would come next. A recurrent neural network, however, can remember those characters because of its internal memory. It produces output, copies that output and loops it back into the network.

Simply put: Recurrent neural networks add the immediate past to the present. Therefore, a RNN has two inputs: the present and the recent past. This is important because the sequence of data contains crucial information about what is coming next, which is why a RNN can do things other algorithms cannot. A feed-forward neural network assigns, like all other deep learning algorithms, a weight matrix to its inputs and then produces the output. Note that RNNs apply weights to the current and to the previous input. Furthermore, a recurrent neural network will also tweak the weights for both gradient descent and backpropagation through time (Singh, 2020).

#### 4.4. RNN topologies

We have seen examples of how MLP and CNN architectures can be composed to form more complex networks. RNNs offer yet another degree of freedom, in that it allows sequence input and output. This means that RNN cells can be arranged in different ways to build networks that are adapted to solve different types of problems. Figure 8 shows five different configurations of inputs, hidden layers, and outputs, represented by red, green, and blue boxes respectively: Of these, the first one (one-to-one) is not interesting from a sequence processing point of view, since it can be implemented as a simple Dense network with one input and one output. The one-to-many case has a single input and outputs a sequence. An example of such a network might be a network that can generate text tags from images, containing short text descriptions of different aspects of the image. Such a network would be trained with image input and labeled sequences of text representing the image tags (Karpathy, 2015). Below are the several types of RNN usage (Roman, 2020).

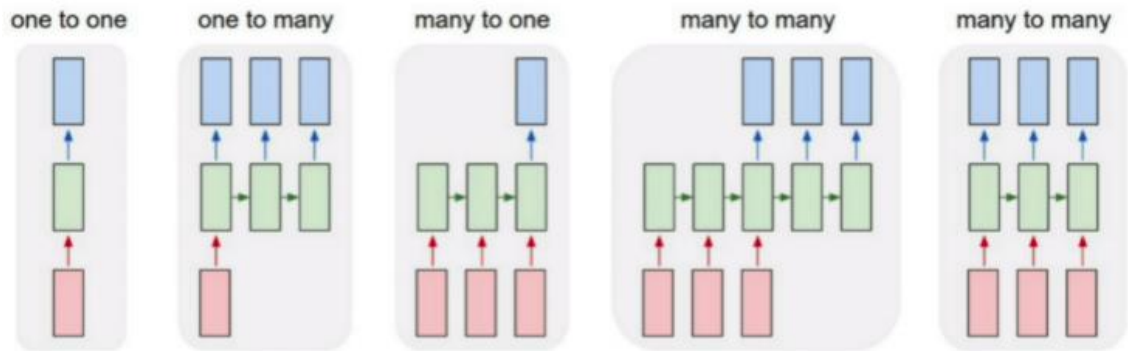


Figure 8: Common RNN topologies

More information on the process graphs of the above graphic is provided below. The arrows denote functions, and each rectangle is a vector (e.g., matrix multiply). Green vectors hold the state of the RNN while red vectors represent input, blue vectors represent output (more on this soon). (1) Vanilla mode of processing without RNN, from fixed-sized input to fixed-sized output, from left to right (e.g., image classification). (2) Output in sequence (e.g., image captioning takes an image and outputs a sentence of words). Input for the sequence (e.g., sentiment analysis where a given sentence is classified as expressing positive or negative sentiment). (4) Sequence input and sequence output (e.g., Machine Translation: an RNN reads a sentence in English and then outputs a sentence in French). (5) Synced sequence input and output (e.g., video classification where we wish to label each frame of the video). Notice that in every case are no pre-specified constraints on the lengths sequences because the recurrent transformation (green) is fixed and can be applied as many times as we like.

Generally:

### **One-to-one**

With one input and one output, this is the classic feed-forward neural network architecture.

### **One-to-many**

This is referred to as image captioning. We have one fixed-size image as input, and the output can be words or phrases of varying lengths.

### **Many-to-one**

This is used to categorize emotions. A succession of words or even paragraphs of words is anticipated as input. The result can be a continuous-valued regression output that represents the likelihood of having a favorable attitude.

### **Many-to-many**

For machine translation, like that used by Google Translate, this paradigm is appropriate. A sentence in English of any length may be the input, and a sentence in English of any length could be the output. The final many to many models can be used for video classification on a frame-by-frame basis. As you may already be aware,

traditional RNNs struggle to accurately capture long-distance dependencies. This mainly has to do with the issue of vanishing gradients. Gradients or derivatives diminish exponentially as they move down the layers while training very deep networks. The problem is referred to as the Vanishing Gradient Problem.

To tackle the vanishing gradient the LSTM (long-short-term-memory) was introduced as its name derives from the problem. The RNN hidden layer is modified with LSTM. RNNs can remember their inputs for a long time thanks to LSTM. In LSTM, a cell state is transferred to the next time step in addition to the concealed state (LENDAVE, 2021). Below, on figure 9, we will have a look at the architecture of the LSTM (Roman, 2020).

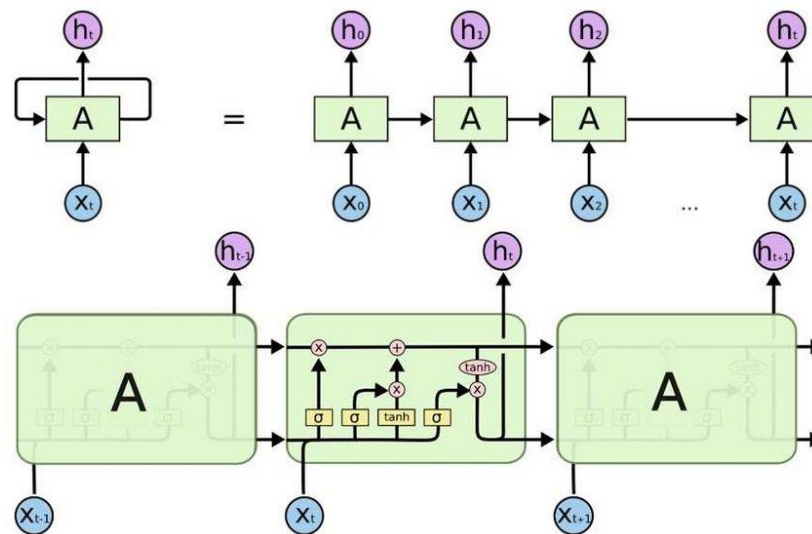


Figure 9: RNN architecture

Through LSTM, long-range dependencies can be captured. It has a long-term memory for previous inputs. There are three gates in an LSTM cell. In the LSTM, these gates are employed for memory manipulation. Gates in long short-term memory regulate the gradient propagation in the memory of a recurrent network. For sequence models, LSTM is a common deep learning technique. The LSTM algorithm is employed in real-world applications such as Apple's Siri and Google's voice search, and it is responsible for their success (Choubey, 2020).

#### 4.5. Sequence Data

However, in many situations, such as with language, voice, and time-series data, one data item is dependent on those that come before or after it. Traditional machine learning assumes that data points are distributed independently and identically. Another word for this kind of information is sequence data. Machine learning also uses a similar concept of sequencing to learn from a sequence of data. The paragraphs that follow will discuss sequential machine learning. We will also go over the numerous sequential machine learning models that are employed, as well as how sequential data is used in modeling. When the dataset's points are dependent on one another, the data is referred to as sequential. A Time-series is a common example of this, with each point reflecting an observation at a certain point in time, such as a stock price or



sensor data. Sequences, DNA sequences, and meteorological data are examples of sequential data (Jayawardhana, 2020).

In other words, we can term video data, audio data, and images up to some extent as sequential data.

Below are a few basic examples of sequential data, that I have listed some popular machine learning applications that are based on sequential data:

- Time Series: a challenge of predicting time series, such as stock market projections.
- Text mining and sentiment analysis are two examples of natural language processing (e.g., Learning word vectors for sentiment analysis)
- Machine Translation: Given a single language input, sequence models are used to translate the input into several languages.
- Image captioning is assessing the current action and creating a caption for the image.
- Deep Recurrent Neural Network for Speech Recognition Deep Recurrent Neural Network for Speech Recognition
- Recurrent neural networks are being used to create classical music.
- Recurrent Neural Network for Predicting Transcription Factor Binding Sites based on DNA Sequence Analysis

To efficiently model with this data or to get as much information, it contains a traditional machine algorithm that will not help as much (Lai, 2021).

## 4.6. The Sequence Learning

Sequence models are machine learning models that input or output data sequences. The invention of Sequence Models was driven by the study of discrete sequential data, including time-series, text phrases, and other sequential data. Convolutional Neural Networks are more adapted to manage spatial data, whereas these models are better suited to handle sequential data. The crucial element to remember about sequence models is that the data we are working with are no longer independently and identically distributed samples, and the data are reliant on one another due to their sequential order. As mentioned above, for speech recognition, voice recognition, time series prediction, and natural language processing, sequence models are particularly popular (Shrott, 2019).

### 4.6.1. Sequence-To-Sequence

Sequence models are machine learning models that input or output data sequences. The invention of Sequence Models was driven by the study of discrete sequential data, including time-series, text phrases, and other sequential data. Convolutional Neural Networks are more adapted to manage spatial data, whereas these models are better suited to handle sequential data.

By taking two inputs at each moment in time, it constructs the context of the word. The name recurrent comes from the fact that it receives two inputs, one from the user and the other from the previous output (output goes as input) (Dietterich, 2004).

Figure 10 shows Seq2Seq Architecture. The model consists of two parts: an encoder (left) and a decoder (right) (Jeon, 2019).

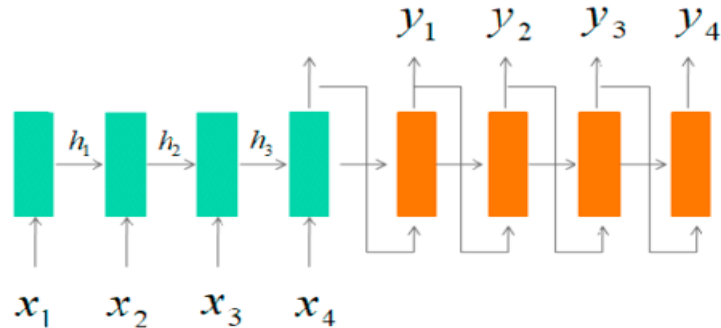


Figure 10: LSTM Encoding-decoding

#### 4.7. Long Short-Term Memory (LSTM)

RNN extensions that expand the memory are known as long short-term memory (LSTM) networks. The foundational units for an RNN's layers are LSTM. By giving data "weights," LSTMs enable RNNs to either accept new information, forget it, or give it enough weight to affect the result. Recurrent neural networks are extended by long short-term memory networks (LSTMs), which essentially expands memory. As a result, it is suitable for learning from significant events that have very large delays between them.

The layers of an RNN, which is frequently referred to as an LSTM network, are constructed using the units of an LSTM. RNNs can recall inputs for a long time because to LSTMs. This is due to the fact that LSTMs have a memory that stores information, much like a computer's memory. The memory of the LSTM can be read, written to, and deleted. This memory can be viewed as a gated cell, where the cell selects whether to store or erase information based on the value it attributes to the information (i.e., if it opens the gates or not). Weights, which the algorithm also learns, are used to determine importance. This merely indicates that it gradually comes to understand what information is crucial and what is not. There are three gates in a long short-term memory cell: an input gate, a forget gate, and an output gate. These gates decide whether to allow additional input (input gate), erase the data because it is unimportant (forget gate), or allow the information to affect the output at the current timestep (output gate). An example of an RNN with its three gates is shown in the below figure 11 from (Dires Negash Fente, 2018):

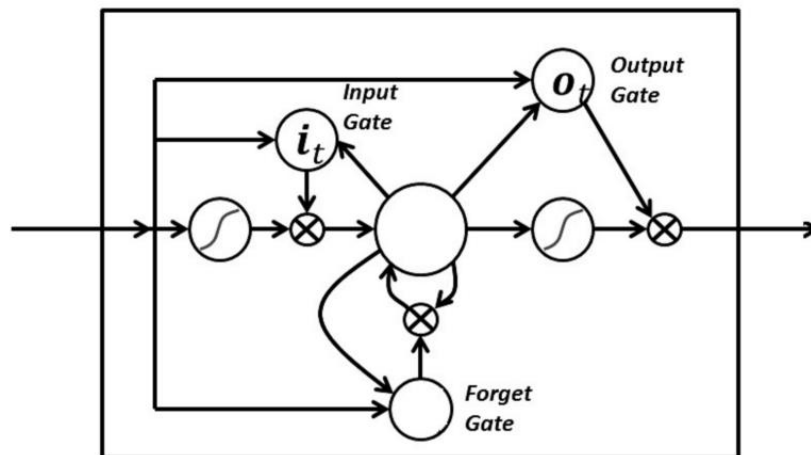


Figure 11: Gates and flow in RNN

The control flow of an LSTM is comparable to that of a recurrent neural network. As it propagates forward, it processes data and transmits information. The actions taking on within the LSTM's cells are what differ. The LSTM has complex architecture as presented in the below figure 12 from (Irshad, 2020).

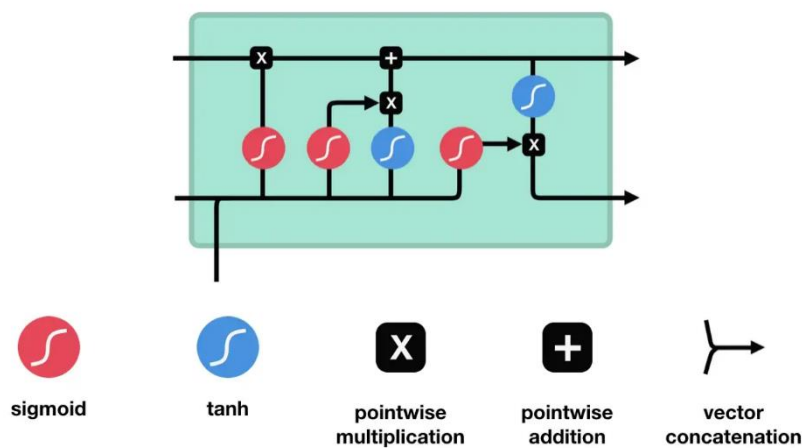


Figure 12:LSTM's cell and operations\

These operations are used to allow the LSTM to keep or forget information. Now looking at these operations can get a little overwhelming, so we will elaborate step by step.

The cell state and its multiple gates make up the fundamental idea of LSTMs. The cell state functions as a highway for the transportation of relative information throughout the entire sequence chain. It can be regarded as the network's "memory". The cell state might, in theory, carry important information when the sequence is processed. The effects of short-term memory are thus diminished because even knowledge from

earlier time steps might travel to later time steps. Information is added to or withdrawn from the cell state via gates as the cell state travels. The gates, which determine which information is permitted on the cell state, are various neural networks (Bediako, 2017).

Sigmoid activations can be found in gates. The tanh activation is comparable to a sigmoid activation. It compresses data between 0 and 1 rather than between -1 and 1. Because any integer multiplied by 0 is 0, values vanish or are "forgotten," making it easy to update or forget data. Any number multiplied by one has the same value, hence that value is "preserved" or remains the same. The network can learn which data is crucial to maintain and which should be deleted based on importance.

Now, we will describe the meaning of gates. The forgotten gate comes first. This gate determines what data should be deleted or retained. The sigmoid function processes data from the previous hidden state as well as data from the current input. There are values between 0 and 1. To forget means getting closer to 0, and to keep means getting closer to 1. We have the input gate to change the cell state. First, we feed a sigmoid function the current input and the prior concealed state. By converting the values to be between 0 and 1, it determines which values will be changed. 1 denotes importance, 0 indicates it is not important. Additionally, the hidden state and current input are passed to the tanh function, which squashes values between -1 and 1 to aid in network regulation. The tanh output is then multiplied by the sigmoid output. Which information from the tanh output should be retained will be determined by the sigmoid output. Now that we know more, we should be able to determine the cell state. The forget vector is first multiplied pointwise by the cell state. If multiplied by values close to 0, this could result in the cell state losing values. Then, using a pointwise addition, we update the cell state to new values that the neural network deems pertinent by taking the output from the input gate. Now we have our new cell state. The output gate comes last. What should be the next concealed state is determined by the output gate. Keeping this in mind, the concealed state comprises data about prior inputs. For forecasts, the concealed state is also utilized. We start by feeding a sigmoid function the current input as well as the prior concealed state. Then, we provide the tanh function the newly updated cell state. The information the hidden state should contain is determined by dividing the tanh output by the sigmoid output. The concealed state is what is produced. Following that, the new cell state and new concealed are carried over to the following time step. The Forget gate determines whether information should be retained from earlier processes for review. The input gate determines what data from the current phase should be added based on its relevance. The next hidden state is decided by the output gate (Zadransk'a, 2019).

#### **4.8. Why to use LSTM**

RNN's LSTM are utilized to circumvent the vanishing gradient issues that come with training vanilla RNNs with BPTT. The LSTM RNN is able to overcome the BPTT issue by making sure that a consistent error is maintained. This allows the RNN to learn over a long period of time, allowing it to distantly associate problems and their effects. This is what the LSTM's gated cell allows it to accomplish. This gated cell has two states—open and closed—and because it decides what data may be written to, read from, and stored on it, acts like a computer's memory. This feature is what enables it to keep details of attacks learnt from training process and make detection decisions

based on this stored information on gated cell. However, the difference to computer memory is that this gate is analog and not digitized.

The majority of the preprocessing and modeling tasks while executing standard text modeling are focused on creating data sequentially. Examples of similar tasks include text sequencing, stopwords deletion, and POS labeling. These techniques aim to make data easier for a model to understand in accordance with the established pattern. It may provide the outcomes (Duran, 2020).

Applying LSTM networks in this situation may have a unique advantage. We explained how LSTM includes a feature that allows it to memorize the order of the data earlier in the essay. The text data always contains a lot of unused information, which can be removed by the LSTM to shorten calculation times and lower costs, therefore this function also works on the elimination of unused information. The LSTM is a strong tool for text categorization and other text-based tasks since it has the ability to remove unnecessary information and memorize the order of the information. There are various tasks we can perform in the time series analysis domain using LSTM.

#### 4.9. Why LSTM outperform RNN

We will first examine an RNN and an LSTM side by side. RNNs' amazing performance in sequence learning has been amply demonstrated. But it has been remarkably observed that RNNs do not handle long-term dependencies with much panache. The below figure 13 from (Robail Yasrab M. P., 2020) presents the differences between RNN and LSTM. On the left RNNs use their internal state (memory) to process sequences of inputs, while on the right Long Short-Term Memory (LSTM) network is a variant of RNN, with additional long-term memory to remember past data.

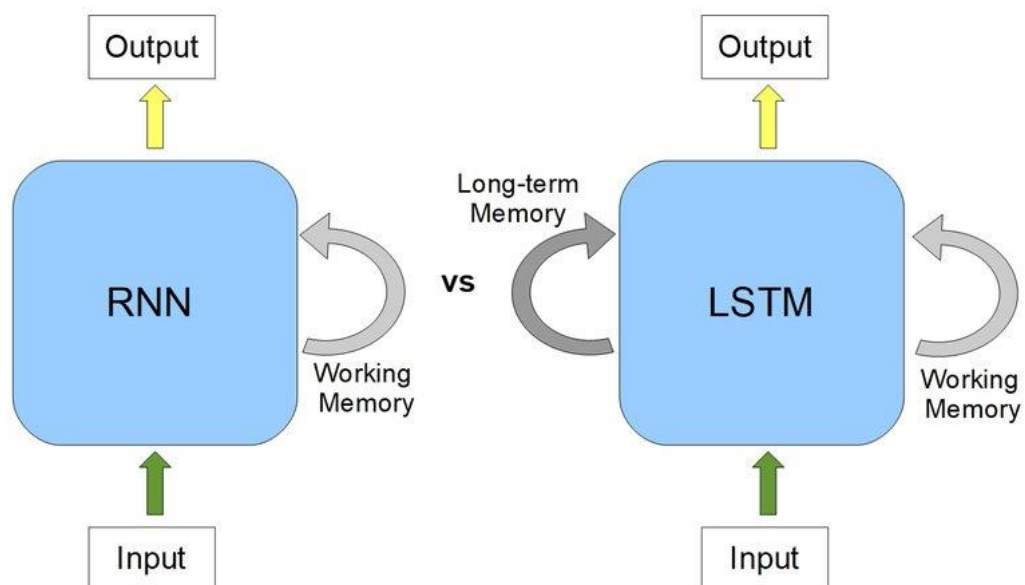


Figure 13: Working of Recurrent Neural Network vs LSTM

Why so? The tanh function, also known as a hyperbolic tangent function, is used by recurrent neural networks. This activation function's range is  $[-1, 1]$ , and the range of its derivative is  $[0, 1]$ . RNNs are now understood to be a type of deep sequential neural network. As a result, the network's matrix multiplications keep getting deeper due to its depth as the input sequence gets longer. As a result, even when we apply the chain rule of differentiation to backpropagation calculations, the network continues to multiply large values by small ones. And guess what happens if you continue to multiply a negative integer by itself? It shrinks exponentially until the final gradient is almost zero, at which point model training stops and weights are no longer updated. When we talk about RNNs, it results in poor learning, which we describe as "cannot handle long-term dependencies" (Mjaaland, 2020). Let's assume that in this scenario, our gradient value is more than 1, and multiplying a huge number to itself makes it exponentially larger resulting to the explosion of the gradient. This is a similar notion to the vanishing gradient problem, but merely the opposite of the process.

RNNs use their internal state (memory) to process sequences of inputs, b: Long Short-Term

Memory (LSTM) network is a variant of RNN, with additional long-term memory to remember past data. The aforementioned issue was solved and the time dependences in the provided data were captured using LSTMs. By retaining and learning large time-based dependencies in data, LSTM provides a solution for exploding/vanishing gradients in addition to resolving temporal dependencies. In comparison to CNNs, LSTM is simpler to manage, train, and fine-tune. These can be successfully used with CNN to provide a static-dynamic machine learning combo, which can be used to tackle common classification issues using temporal data (Robail Yasrab M. P., 2020).

## 5. Customer Behavior Modeling

### 5.1. Experiment 1

The experiment 1 aimed to predict the next customer activity based on a specific sequence-pattern. To make it clear, in this experiment we tried to predict the upcoming, next activity – activities given a specific pattern of activities happened before. It is the way of predicting the most probable activity following a series of activities based on them. As mentioned before, it is one of the fundamental exercises that customer centric companies try to figure out. Following we will elaborate how we worked on this.

#### 5.1.1. Data Pre-process

As explained on the above chapters, customers behavior prediction is not an easy thing. There are many features that affect on it. So, it would be a glib approach to use the whole dataset for such an experiment. The analysis should be focused on groups of customers who have something in common. Here, we worked by splitting the dataset into groups based on the CLT (the ‘when’ he/she left his/her first footprint). In more detail, we focused on the customers who have more than 15 months with the service. Our data would drive into a more meaningful result that way. Although we cut some records, we still have a decent volume of customers ~20K and a well enough volume of activities made by them, almost ~180K. We picked this group as the customers could be more aware of the ecosystem of the company and more experienced with the capabilities with the service. The below figure 14 presents the allocation of length of sequence with the experimental customer data. To explain further, the horizontal axis shows the group of activity sequence, and the vertical presents the number of customers who interacted (made the relevant number of activities) with the service. There are ~1000 customers who made 7 activities on the respective period. The average number of activities is 9.

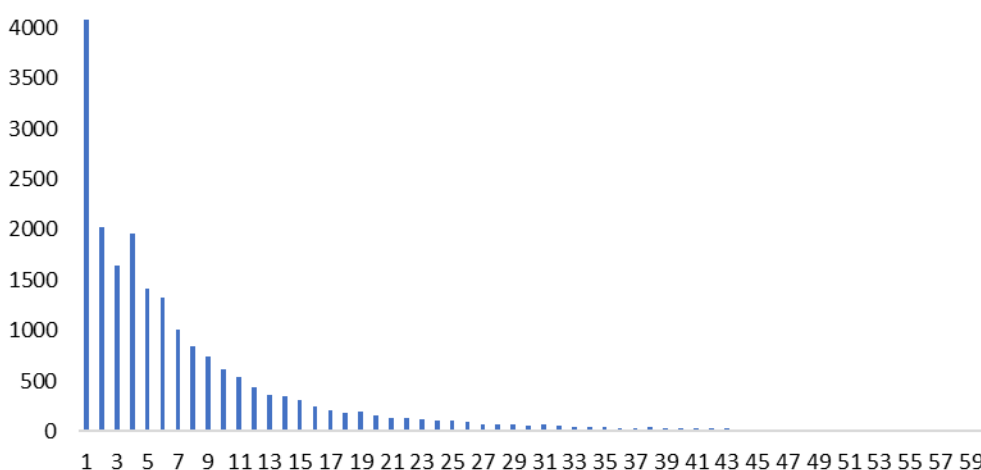


Figure 14: Allocation of Number of Activities that made by Customers. Explanation: ~ 4K Customers with CLT more than 15 months have only 1 interaction with the service.

We will use both layers of activity Type1 and Type2, starting with the least complicated level, the Type1. Worth to mention that the data preprocessing takes more than 50% of the total effort of the completion of the experiment.

Deep Learning models are data-hungry and require a lot of data to create the best model or a system with high fidelity. The quality of data is as important as the quantity even if you have implemented great algorithms for machine learning models. The following quote best explains the working of a machine learning model. Garbage In Garbage Out (GIGO) (Khan, 2022). If we feed low-quality data to ML Model, it will deliver a similar result. Here, we are privileged to have a large amount of data even if we split the dataset to target in specific audience of customers.

On this part we will explain the major steps we followed to reach the results on experiment 1. As mentioned above, we are privileged to filter the dataset to achieve better results. We will not work with the whole dataset here. So, let's see how accurate the prediction of the 'next' activity of the customers could be are using the service for more than a year. This selection could drive into a better training model, something crucial for the end results.

### **5.1.2. Tokenization**

After data processing we started to work with the ML python libraries frameworks as Keras Tokenization. Keras Tokenization is the process in which we provide a unique number to each activity and make an activity index, or we can say sequence of numbers. Tokenization is a key (and mandatory) aspect of working with text data. It is a way of separating a piece of text into smaller units called tokens (Pai, 2022). Here, tokens can be either words, characters, or subwords. Hence, tokenization can be broadly classified into 3 types – word (we used that), character, and subword (n-gram characters) tokenization. Tokenization is used in natural language processing to split paragraphs and sentences into smaller units that can be more easily assigned meaning.

The total number of activities is 8, which means that we are going to use 8 different numbers, each one for each activity. So, every activity could be signed with a specific number from 1 to 8. The null value is signed with the 0.

### **5.1.3. Padding**

The interesting part of coding starts here. With padding we tried to make all sequences by having the same length. The length of every sequence must be the same. To make it, we need to find the sequence that has a maximum length, and based on that length, we must pad rest of sequences. Padding is a special form of masking where the masked steps are at the start or the end of a sequence. Padding comes from the need to encode sequence data into contiguous batches: in order to make all sequences in a batch fit a given standard length, it is necessary to pad or truncate some sequences.

Remember the problem that we have with RNN, the vanishing gradient. So, if you are using Pre-padding, then there are actual values to be remembered by the RNN at the last and for post-padding, the actual values are at the start. Hence, if there is long padding, there is a high chance that the model might forget what the actual values are



that need to be remembered and the model might not perform well. So, I personally you to use pre-padding rather than post-padding (Inf, 2020).

```
array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 3, 7, 7, 7, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 7],
      dtype=int32)
```

Figure 14: depiction of an activity-series of a customer

The above array is one the sequences of our dataset. Each activity has been signed with a number. The zeros are signs of non-existed activity. The customer above has made 15 activities in total during the time window. The activities he/she made are 3s or 7s, signed only two types of activities.

#### 5.1.4. Model Training

Long Short-Term Memory (LSTM) networks is an advance recurrent neural network which is capable to store order states by using its cell state feature.

As you may see there are many features in this model. Early stopping is one of them, which stops training when a monitored metric has stopped improving. Early stopping is an important function not only in order to save time in training part but also to avoid overfitting and confuse the model. Adam is an optimization algorithm that can be used instead of the classical stochastic gradient descent procedure to update network weights iterative based in training data. Adam is a popular algorithm in the field of deep learning because it achieves good results fast. Categorical cross-entropy is a loss function that is used in multi-class classification tasks. These are tasks where an example can only belong to one out of many possible categories, and the model must decide which one. Formally, it is designed to quantify the difference between two probability distributions. One Epoch is when an entire dataset is passed forward and backward through the neural network only once. Since one epoch is too big to feed to the computer at once we divide it in several smaller batches. Maybe in an experiment like this the number of epochs should be around 40-50, but we cut it earlier as the model did not look like being improved. It is worthy to be said once more that the filtered dataset worked better the whole hear. A group of customers having the same CLT make us to attend common behavior.

Following, you may find the evolution charts of loss and accuracy through the epochs. The results are good for such a complicated problem that we are trying to approach. We trained the model for just 15 epochs, although the output is a bit promising for

more, even for accuracy metric or loss. The lines of accuracy and loss behave similarly on the first 3 epochs, as they have extreme move before being normalized.

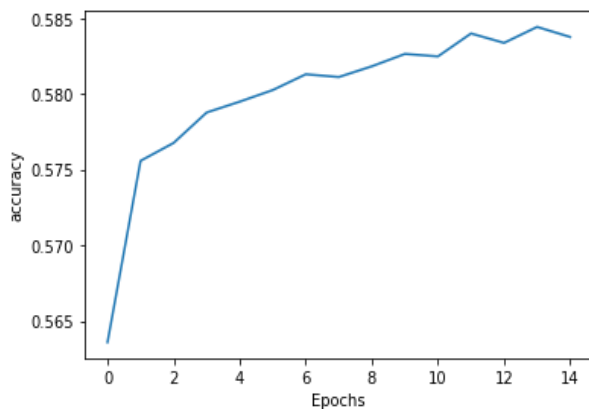


Figure 15: accuracy score by epoch

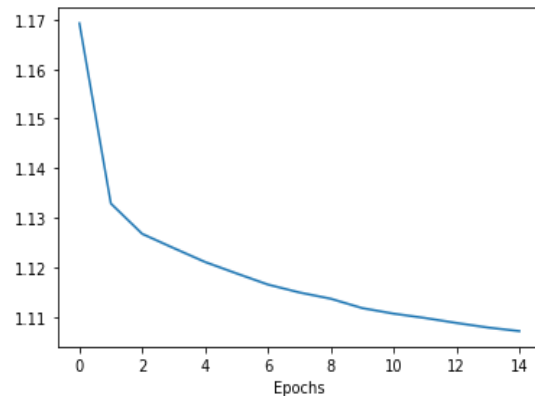


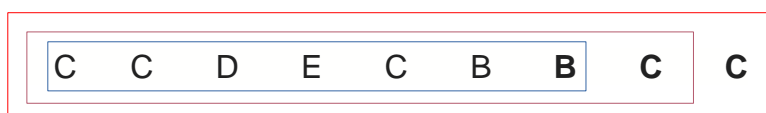
Figure 16: loss score by epoch

Finally, our model reached ~58% accuracy. Almost 6 out of ten predictions of the upcoming activities will be correct. Let's go to the final part to make some predictions with this model. Importing some a sequence of activities, we can predict the upcoming activity or activities. Meaning that given 4 activities 'already done', I can predict the upcoming 1+n activities, as the model will recalculate the new last activity of the new generated sequence. To make it clear, see below the action.

Given the following sequence of activities "C C D E C B" and querying the next 3 upcoming actions, our model makes the following prediction taking into consideration each new sequence that takes shape.

Results:

1/1 [=====] - 0s 25ms/step  
 1/1 [=====] - 0s 24ms/step  
 1/1 [=====] - 0s 23ms/step



Predicting a future is not a dream anymore, cause AI is there with us. These results are valuable for the marketing team once they would like to target specific audiences. Imagine a marketing campaign targeted to each group of customers that supposed to behave in a certain way. The revenue of the company could have potential to increase. Customers could be happy by feeling individual appealing approach from the company that they have chosen. The above experiment has derived from Type1 level of activity. What if we execute the same code with a more detailed dataset? Let's try the above by working with Type2 column.

### 5.1.5. Working on a Granular Level

As we said before, the dataset used is crucial for the training of the model and the final prediction results. At this point, we used the same group of customers as before, but we focused on detailed activity type column (Type2). To avoid any misunderstandings, see the image below.

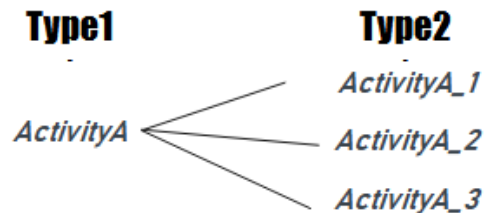


Figure 17: hierarchy level of activity type (Type1 to Type2)

Following the same steps with the previous experiment, below we can see that the changes we made in our data entry, affected the model and the results. The most important factor is that the number of activities is 23 vs 8 that was before. In the real world, such a level of detail is not highly recommended to be taken into consideration from the Data Scientists team. The campaigns cost a lot of money and the possibilities to communicate the wrong messages to the wrong customers could have negative results. The Type1 level of analysis sounds for feasible for a marketing campaign project. Moreover, we can use Type2 for secondary analysis or for further action in special cases.

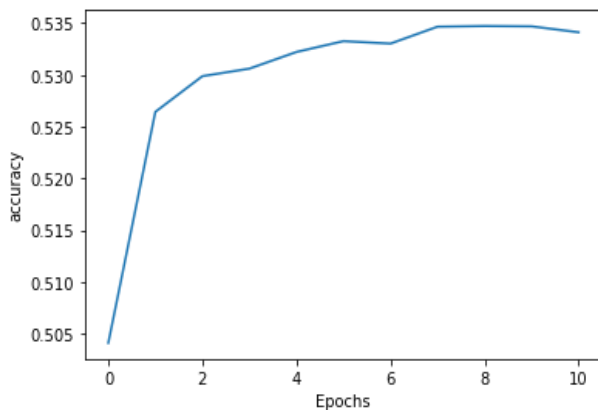


Figure 19: accuracy score by epoch (Type2)

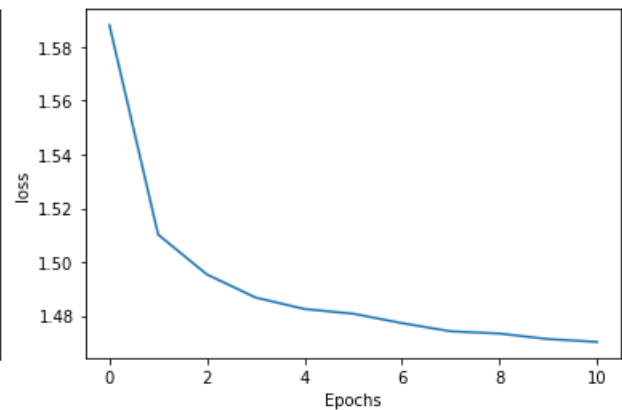


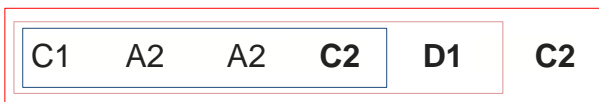
Figure 18: loss score by epoch (Type2)

Moving to the tech part now, the training model parameters are exact the same with the previous attempt (Type1) but due to the increased number of observations, we reached accuracy around 53%. The accuracy rate decreased by 5pp. Following, you may find the evolution charts of loss and accuracy through the epochs. The trends are similar with the 'Type1' results, while the end results have lower performance. Using the early stopping function the training process stopped on 11<sup>th</sup> epoch. As we can see from the charts, the accuracy and loss metrics decreased after the 10<sup>th</sup> epoch.

Knowing what you want to predict will help you decide which data may be more valuable to collect. When formulating the problem, conduct data exploration and try to think in the categories of classification, clustering, regression, and ranking that we talked about in our whitepaper on business experiment of machine learning. In the real world, this could be the chosen model for a more personalized CRM campaign. The aim is to communicate the right customers, so, by understand in more detail their behavior you could make them feel individual. It is tempting to include as much data as possible, because of... well, big data! That is wrong-headed. Yes, you want to collect all data possible. But if you're preparing a dataset with particular tasks in mind, it's better to reduce data. Since you know what the target attribute (what value you want to predict) is, common sense will guide you further. You can assume which values are critical and which are going to add more dimensions and complexity to your dataset without any forecasting contribution. This approach is called attribute sampling (Kenton, 2021). For instance, adding bounce rates may increase accuracy in predicting conversion.

Given the sequence of "C1 A2 A2" and querying the next 3 upcoming actions, our model makes the following prediction taking into consideration each new sequence that takes shape.

1/1 [=====] - 0s 24ms/step  
 1/1 [=====] - 0s 21ms/step  
 1/1 [=====] - 0s 20ms/step



We started off by talking about how important it is to have a dataset that meets a certain standard of quality, as without setting a boundary for quality, your model is severely handicapped and cannot do its job properly. In a nutshell, keeping a check on your dataset quality and continuously looking for and implementing ways to improve is what makes you effectively solve a problem through ML or AI.

ML/DL application is not only about dealing with data. It is also to give some advice when it comes to interpreting the data to improve the business processes. It is quite challenging to link an idea about data science, machine learning and artificial intelligence to a marketing strategy. Any parameter in the ML/DL application, from the dataset until the libraries and the factors, is crucial and demands strong feeling of how to adapt it to a particular strategy.

Our work using the Type2 level of detail is not as accurate as on the Type1 level, and that is ok. Even with the first approach, we are able to create many different segments of customers and to initiate some targeted campaigns. From business and scientific perspective, without a doubt, we have to proceed with the Type1 approach.

## 5.2. Experiment 2

Moving on with a different approach. This approach will be more complicated than the previous one. With the aim to achieve a better accuracy and provide to the potential stakeholders with a more targeting campaign, we took a stab on this with a different way of working. With the same goal, to predict next customer activity, we try a new experimental way of working by focusing on the last 4 activities that the customer made. So, what is the last one, the 5<sup>th</sup>? On the upcoming steps, we will mention the way of working.

### 5.2.1. Data Pre-processing

Obviously, on this part we had to make a different data pre-processing. For this experiment, we need to create a different data structure as we are interested in only the last 5 activities. As the topic of this thesis is not the data analysis, we will not elaborate in detail about the preprocessing steps. We used the whole dataset on this experiment, using all customers without restrictions. The first filtering was to cut off the 2 types of activities that seem to confuse the model on the training part. So, we have a new dataset that contains the customers who made the specific 6 activities and not the 2 which caused confusion. As in the first experiment, this experiment is executed with the Type1 level of detail.

Briefly, the steps we followed:

- calculate the total number of activities by customer, in order to find the total volume of records. It helps to identify the loss of data after the upcoming filtering.
- keep the last 5 activities for the customers and cut off the customers who have less than 5 activities. Here, we rechecked if the remaining dataset is still decent for the ml application.
- calculate the date diff between two consecutive activities. We will use the date diff between each activity as a parameter in our model.
- calculate the date diff between first and last activity per customer (min max date of activity). This is for the tenure calculation. (CLT)
- normalize time diff between successive actions, now the values are between 0 and 1, necessary for NN. Normalization is a technique often applied as part of data preparation for machine learning. The goal of normalization is to change the values of numeric columns in the dataset to use a common scale, without distorting differences in the ranges of values or losing information. Normalization is also required for some algorithms to model the data correctly (PeterCLU, Microsoft, 2021).

The dataset looks like the below table:

Customer_ID	CustomerFirst Purchase	TotalTrans	DateDiffNorm Last1	DateDiffNorm Last2	DateDiffNorm Last3	DateDiffNorm Last4	DateDiffNorm Last5	Type1Last1	Type1Last2	Type1Last3	Type1Last4	Type1Last5	Tenure
67966SD	2022-04-13	69	0	0	0.76	0.24	0.14	B	B	A	C	A	209
4576FT	2022-02-16	174	0.25	0.5	0.25	0.25	2.25	A	A	A	A	A	265
66111GV	2022-02-07	24	0	0	0.5	0.5	0	B	B	D	E	E	274
8900MN	2022-03-15	14	0	0	0.66	0.34	0	E	E	D	E	E	238
5553SA	2022-04-27	21	0.5	0	0.5	0.5	0	E	D	A	D	D	195
...	...	...	...	...	...	...	...	...	...	...	...	...	.....
55437MM	2022-03-31	11	0.52	0	1	0	0	G	D	B	B	B	222
95898PP	2022-03-31	16	1.14	1	0	0	0	G	B	B	B	B	222
43678LP	2022-03-31	6	0.22	0.39	0.61	0	1.78	C	C	B	B	C	222
1976BG	2022-03-31	6	0	0	0	0	0	B	B	B	B	D	222
45779NV	2022-03-31	11	0.68	0	0.26	0.74	0.05	B	B	G	E	C	222

Table 3: preview of the final dataset

### 5.2.2. Data Partitioning

The whole data available into two or three non-overlapping sets (the training set, and the test set). If the data set is very large, often only a portion of it is selected for the partitions. The basic idea of data partitioning is to keep a subset of available data out of analysis, and to use it later for verification of the model. Data partitioning is normally used in supervised learning techniques in data mining where a predictive model is chosen from a set of models, using their performance on the training set as the validation of choice. Some examples of such techniques are classification trees, regression trees, neural networks, nonlinear variants of the discriminant analysis. Data partition improves scalability, reduces connections, and optimizes the performance. Data partitioning in data mining is the division of the whole data available into two non-overlapping sets: the training set and the test set. If the data set is very large, often only a portion of it is selected for the partitions (PeterCLu, Partition and Sample component, 2021). Partitioning is normally used when the model for the data at hand is being chosen from a broad set of models. The basic idea of data partitioning is to keep a subset of available data out of analysis, and to use it later for verification of the model. Here we split into training and test set using the 80% of our available data for the training dataset and the 20% for the test dataset. Usually, the split could be 65-35 but because of the cut of records for this experiment, as we kept the last 5 activities of each customer, we increased the contribution of the training dataset to achieve better results.

### 5.2.3. Min-Max Scaling

Min-Max scaling is a normalization technique that enables us to scale data in a dataset to a specific range using each feature's minimum and maximum value. The scale of data for some features may be significantly different from those of others, which may harm the performance of our models. It is especially the case with algorithms that rely on a measure of distances, such as Neural Networks and KNN (hemavatisabu, 2018). It transforms data by scaling features to a given range. It scales the values to a specific value range without changing the shape of the original distribution. We will use it for 'Tenure' (CLT) and 'Total Trans' features. We used the range of 1-10 to scale the relevant data.

#### **5.2.4. One Hot Encoding**

Machine learning and deep learning models, like those in Keras, require all input and output variables to be numeric. This means that if your data contains categorical data, you must encode it to numbers before you can fit and evaluate a model. One-hot encoding ensures that machine learning does not assume that higher numbers are more important. As a machine can only understand numbers and cannot understand the text in the first place, this essentially becomes the case with Deep Learning & Machine Learning algorithms. One hot encoding can be defined as the essential process of converting the categorical data variables to be provided to machine and deep learning algorithms which in turn improve predictions as well as classification accuracy of a model (Dey, 2021). One Hot Encoding is a common way of preprocessing categorical features for machine learning models. This type of encoding creates a new binary feature for each possible category and assigns a value of 1 to the feature of each sample that corresponds to its original category. For example, the value '8' is bigger than the value '1', but that does not make '8' more important than '1'. That is exactly what we have to do. Now, we used one hot encode categorical and transform into arrays all variables. We managed to keep variable separated for investigation during modeling. After the encoding, we reshaped our data to be compatible with NN input shapes. The same process is repeated for train and test sets.

#### **5.2.5. Built the Model**

Training the model is hands down the most time consuming and expensive part of machine learning. Training the model on a GPU can give speed gains close to 40x, taking 2 days and turning it into a few hours. However, this normally comes at a cost to our wallet. The time saving and the expensive GPU options make to look for another way and that is how I found a great tool called Google Colab. I would describe Colab as the google docs equivalent of Jupyter notebooks. Colab is aimed at being an education and research tool for collaborating on machine learning projects. The great part is, that it is completely free forever (Bourdakos, 2018). There is no setup to use it. I did not even need to log in. The best part is that you get an unlimited supply of 12 hours of continuous access to a k80 GPU, which is powerful stuff.

In coding part, we used a lot of frameworks, packages and modules in this experiment given it was a complex solution. The most important are from Tensorflow, Keras and numpy. We used the LSTM for time difference between actions. The rest data are concatenated before dense layer that predicts the 5th activity. To specify the model using the LSTM cells in recurrent neural networks, we define the dimensions for parameters as the gender, age, the number of transactions, the tenure of each customer. In compilation part, even we made many attempts and experiments, we concluded on usage of Adam optimizer for our Keras model (Jason, 2017). The Adam optimization algorithm is an extension to stochastic gradient descent that has recently seen broader adoption for deep learning applications in computer vision and natural language processing.

When a certain number of samples are processed, the model is updated. This is referred to as the sample batch size. The number of complete passes in the training dataset is equally significant and is referred to as the epoch in machine learning. Our

model ran for 40 epochs having a better accuracy than the previous 2 models. Following the results of the 40<sup>th</sup> epoch. As we expected that approach is more accurate than the previous experiment (5.2). The filtered dataset is an explanation for this. For sure, unless we choose this dataset, we would have different results.

Epoch 40/40

58/58 [=====] - 1s 11ms/step - loss: 1.0803 - accuracy: 0.6228 - val\_loss: 1.0949 - val\_accuracy: 0.6128

Graphs and visualization always help us to connect the dots. So, below you may find the relevant accuracy and loss charts, comparing test with train dataset on validation metrics.

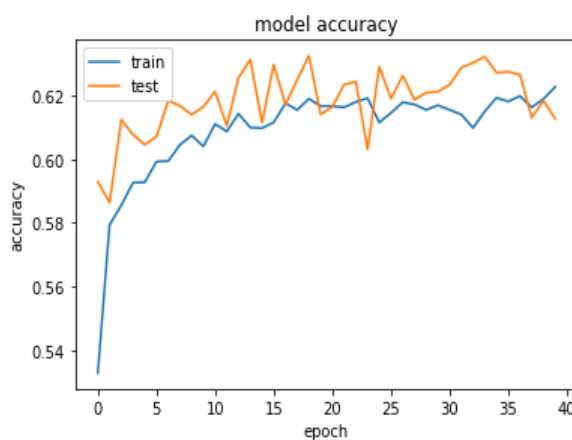


Figure 21: accuracy score by epoch for test and train set

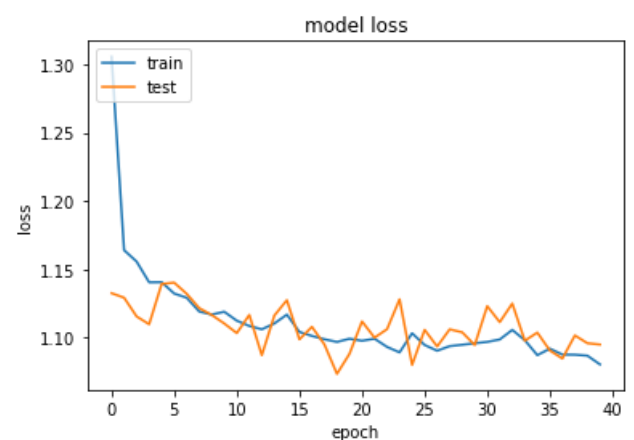


Figure 20: loss score by epoch for test and train set

In both charts the presented lines are the proof that our model has a nice fit. Train and Test have the same trend in both metrics through the training process. Both lines met and came together.

Time to put the model to the test part! Let's evaluate how our model has performed with its predictions on the test data. We use the `model_evaluation_utils` module for these which leverages scikit-learn internally to give us standard classification model evaluation metrics. Following, in the below table 4, you may find the evaluation metrics matrix, per class for the test set. We focus on f1, which is quite good for such a complex query. F1 Score might be a better measure to use if we need to seek a balance between Precision & Recall and there is an uneven class distribution (large number of Actual Negatives).



	precision	recall	f1-score	support
A	0.70	0.72	<b>0.71</b>	2639
B	0.67	0.22	<b>0.33</b>	1247
C	0.64	0.76	<b>0.70</b>	1733
D	0.60	0.53	<b>0.56</b>	686
E	0.25	0.22	<b>0.23</b>	266
F	0.53	0.75	<b>0.62</b>	1383
<b>accuracy</b>			<b>0.62</b>	7954
macro avg	0.56	0.53	0.53	7954
weighted avg	0.63	0.62	0.60	7954

Table 4: Model evaluation matrix

And that is, a good F1 score means that you have low false positives and low false negatives, so you are correctly identifying real threats and you are not disturbed by false alarms. An F1 score is considered perfect when it is 1, while the model is a total failure when it's 0. ActivityA and ActivityC have the best f1-scores within all activity options. The worst score is for ActivityE and ActivityB. Unless the E and B types of activities were important, we could cut them too. Maybe, an extra cut of activity options could drive into a better f1-score overall.

### 5.3. Experiment 3: Customer Churn

In today's commercial world, competition is high, and every customer is valuable. Understanding the customer behavior is of utmost importance. With this customer churn classification application, we try to identify whether customer leaves the commercial business and takes their money elsewhere. Understanding customer churn is vital to the success of a company and a churn analysis is the first step to understanding the customer. The process of computationally identifying and categorizing activity patterns, meaning sequences of customer activities, especially to determine whether the customer's behavior towards him/her to a vulnerable or a secured user customer segment. When predicting churn, you are not just identifying at-risk customers, you're also identifying pain points leading up to churn and helping to increase overall customer retention and satisfaction (Rithika, 2022). Predicting churn helps in preventing churn. Preventing churn, in turn, serves as a great revenue source for businesses.

In this thesis, we flag the customers who were active, means that they have activity presence during the last 4 months, as '0' (active users = non churners). On the other hand, we flag as churners those who have not activities during the above period.

Analyzing all that harvested data using a Neural Network will enable the organization to build profiles of their customers based on the data they have available (John, 2022). Once a bunch of users are categorized, any company can decide what action to take if a customer is highlighted a suspected churn. e.g., they can understand which customers they want to keep — and offer these guys discounts and promotions —

then also identify which customers are ‘lost causes’ and can be let go. The figure 23 (Brownlee, 2014) shows the sequence classification process. Given an input sequence, classify the sequence.

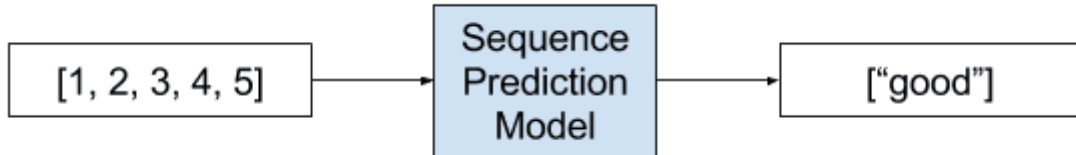


Figure 22: each sequence of activities drives into a specific customer status

This is a totally different way of working than the previous experiments. The main approach of working is to split the dataset into two elements. The one will be the input, meaning the sequence of activities and the second will be the target, meaning the upcoming one. We must transform our dataset in that way before starting with the ML/DL part.

### 5.3.1. Aim

Accurate churn prediction is only as valuable as the effective measures taken to prevent high-risk customers from leaving. So, what do effective measures look like? It is different for every business, and the actions taken will vary based on teams within the organization. It can look like re-engaging a customer through retention emails, reps providing a more personalized experience, re-educating customers on product features, etc. (Dunkley, 2022). The goal of this experiment is to identify if a customer is a retention risked user (churner) or not, based on the sequence of activities that he/she made. We will try to create a model that given a sequence of activities will help us identify if the customer is a possible churner or not. The objective of sequence classification is to build a classification model using a labeled dataset so that the model can be used to predict the class label of an unseen sequence. In simple words, given a sequence of activities, predict whether the customer is churner or not.

### 5.3.2. Data Pre-process

To create your churn model, you need to start with the right dataset. For this experiment we want to focus on a group of customers with similar behavior. So, we assumed that the customers having lifetime more than 9 months with the service, will be a good dataset for our experiment. As potential churners we flagged the customers that did not made any activity in 2022. The dataset includes 6 months of 2022, so it sounds fair to flag someone not having activity in this time window as churner. There are 63K customers for our experiment, with a split 60/40 of potential churners. The level of detail is Type1.

### 5.3.3. Experimental Set-Up

The process of computationally identifying and categorizing sequences of something expressed in a piece of text, especially to determine whether the activity toward a particular customer behavior (churner or active user). Is this customer vulnerable or not? So, in the end of the data preparation, we need to concatenate all activities by customer and create a structured data like the below figure. Meaning that we need to

gather the sequence of activities of each customer which drives into a potential churner or into an active customer.

Type1				Churner
A	B	C	A...	0
A	B	A	A...	1
C	B	B	E...	0
E	B	A	D...	0
D	A	A	C...	0
C	E	E	D...	1
D	A	A	B...	1

Figure 23: dataset core columns

As in the above solutions, tokenization and padding techniques must be implemented here too (5.1.2 – 5.1.3).

### 5.3.4. Model

After the dataset is ready, I compose the LSTM Network. Note that `embed_dim`, `lstm_out`, `batch_size`, `drouput_x` variables are hyperparameters, their values are somehow intuitive, can be and must be played with to achieve good results. Please also note that I am using softmax as activation function. The reason is that our Network is using categorical crossentropy, and softmax is just the right activation method for that. Just like the previous experiments.

Here we train the Network. We should run much more than 5 epochs, but the process took so much time even if it runs on Goggle Colab with GPU. The results are more than decent even with the 5 epochs.

```
Epoch 1/5 1321/1321 [=====] - 871s 658ms/step -
loss: 0.4121 - accuracy: 0.7995
Epoch 2/5 1321/1321 [=====] - 863s 653ms/step -
loss: 0.3917 - accuracy: 0.8089
Epoch 3/5 1321/1321 [=====] - 862s 653ms/step -
loss: 0.3859 - accuracy: 0.8114
Epoch 4/5 1321/1321 [=====] - 860s 651ms/step -
loss: 0.3817 - accuracy: 0.8125
Epoch 5/5 1321/1321 [=====] - 859s 650ms/step -
loss: 0.3768 - accuracy: 0.8137
```

In this experiment we reach the accuracy score of 81%. A very good result for a real-world dataset. Imagine the score that we can reach if we increase the number of epochs. With the aim to identify which pattern of sequence drives into a vulnerable customer let's make two predictions-tests. To test our model, as we did above, we imported the activity sequence of "A E A D" and the prediction was "This user may be a churner". This prediction has 81% accuracy of truth. So, the import towards to churner customer. As a marketing department we must make a special communication and actions to avoid this to happen.

**Results:**

1/1 [=====] - 0s 122ms/step This user may be a churner

On the other hand, when we imported the sequence of "B B A D C" the prediction was "This user is an active user". Meaning that this pattern of activities towards to a retention secured customer. This group of customers need a different perspective of communication, probably focus on care and empathy and not on aggressive sale promos.

**Results:**

1/1 [=====] - 0s 123ms/step This user is an active user

## 6. Visualization with Streamlit

### 6.1. Why to serve the results via WebApp

Whether you love or hate the term “data-driven,” it is something most organizations aspire to become. Essentially, it means a company is oriented to make strategic and tactical decisions based on data rather than opinion or gut instinct. After making a significant investment in an analytics platform that is supposed to make data more accessible, the last thing you want to hear is users are not using it. In a Deloitte study (Dykes, 2021), 67% of managers and executives reported they were not comfortable accessing or using data from their analytics tools. Whether it is a technology fit or training issue, tool usage should not be limited to just analysts and data scientists. To build an inclusive data culture, you need data tools that make the data experience inviting and user-friendly, not imposing, and difficult.

Web-based applications helps the entry users of the ‘data-driven’ philosophy by storing all data in the cloud — on a powerful server that can manage your information and quickly distribute it to users of your software when requested. Servers are also more reliable than individual hard drives, so the chances of your data being lost to hard disk failure are minimized with a Web app. The most important thing is that a user/program owner/stakeholder will be more familiar with a web app than with a BI Tool, as the first exists in many ways into the daily routine while the second is totally new.

Upon the 360 approach, that we follow in the whole thesis, at this point, we will attempt to find the best way (exploitable way) to serve the results to the marketing team. We will work with the Streamlit tool of Python (snehankekke, 2022) to serve the data into a web application. Let us use the output from the experiment 2 (the one with the 4 last activities, in which we predict the upcoming 5<sup>th</sup>). In real world we would use the model to predict the last activity from an updated dataset (contains new customers, new types of activities, extended time if activity happened). For ease of reference, we will use the test dataset of the relevant experiment.

The aim is not to create a common dashboard with charts a KPIs, but to help the marketing team to create group of customers based on the 4 activities made before and the 1 last activity (prediction). The Marketing will be able to filter the dataset based on the activity type of the 1<sup>st</sup>, 2<sup>nd</sup>, 3<sup>rd</sup>, 4<sup>th</sup> and the 5<sup>th</sup> = final prediction. With this, the company will create a very targeted campaign for the customers. (Kösters, 2020) We follow that way to serve the data results as data Visualization unlocks other previously invisible patterns. These other emergent properties in the data can formulate new valuable insights, which could not have been discovered before. Visualization allows business users to recognize relationships and patterns between the data and gives it greater meaning. By exploring these patterns, users can focus on specific areas that need attention in the data, to determine the importance of these areas to move their business forward. Moreover, this attractiveness is achieved by using visually appealing ways of presenting data and adhering to design best practices. Next-level visualizations present data in a very sensible way by using the most appropriate chart and formatting options. In addition, elegant transitions facilitate an attractive and smooth way of moving between different points in the storyline of a visualization. This

will increase a user's engagement with the visualization, thus facilitating easy and quick interpretation and understanding. As a result, the message resonates strongly with the audience. However, it is not as simple as taking the data and placing them in a graph and making it look better. It is an act of balance between the form and a function. A plain graph can be boring to catch the attention or make a point; the most impressive visualization could take away from the data or it could speak volumes. It is important to realize that visuals and data have to work together to convey a message. The interactivity of a Data Visualization can facilitate next-level data exploration that matches a user's specific needs.

Data visualization is an art. There 3 points that could get you closer to the better solution. The first step to creating truly insightful data visualizations is to have a specific question you want to answer with your data. If ever you see a visualization with a distorted critical message, it is highly likely this step was skipped, or the creator was trying to answer multiple questions at one time. To avoid overkill, ensure you have clearly articulated specific questions to be answered with your visualizations (i.e., Is there a relationship between feature A and B?). An effective visualization should achieve two goals: 1) it should clearly reveal the answer to the question it was posed and 2) it should be easy for the audience to quickly grasp what is being presented. A key component for meeting such criteria is to ensure you are selecting the appropriate chart to represent what you wish to reveal. Humans are extremely good at recognizing patterns, but the same cannot be said for our ability to recall. For example, have you ever seen someone who looks familiar, but you have absolutely no idea what their name is? Such scenarios happen to us all because of our design. It is much easier for us to remember someone's face because this correlates with our ability to identify patterns, whereas their name requires us to recall information. With all things said, an effective visualization leans on our natural tendencies to recognize patterns. The use of colors, shapes, size, etc. is an extremely effective technique for emphasizing the most important information you want to display. Data visualization is an art; doing it well involves asking a specific question, selecting the appropriate visualization to use, and highlighting the most important information (Pykes, 2022). The above make us to follow the Streamlit approach for the visualization part of the solution.

## **6.2. Python Streamlit**

Data Science is one of the most trending search topics on google these days. Because of its high demand and exhaustive usage in real-world applications, more and more developers of the community are developing new frameworks and libraries that help data scientists and researchers complete everyday tasks. Streamlit is one of these libraries. With Streamlit, everyone can build data apps in no time. It seamlessly integrates with other python libraries like NumPy, Pandas, Matplotlib, and many more (Pykes, 2022).

Data visualization may be described as graphically representing data. It is the act of translating data into a visual context, which can be done using charts, plots, animations, infographics, etc. The idea behind, is to make it easier for us (humans) to identify trends, outliers, and patterns in data. Data professionals regularly leverage data visualizations to summarize key insights from data and relay that information back to the appropriate stakeholders. For example, a data professional may create

data visualizations for management personnel, who then leverage those visualizations to project the organizational structure. Another example is when data scientists use visualizations to uncover the underlying structure of the data to provide them with a better understanding of their data. And that is exactly what we are trying to do here with python, github and Streamlit.

“How can we make a machine learning script and convert it into an app as simple as possible, so that it basically feels like a scripting exercise?”

— Adrien Treuille (inventor of Streamlit)

Web applications are a great way to display your results for a data science or machine learning project. Developing web applications from scratch requires a lot of time, effort and technical skills. An alternative way to build web applications is to use open-source python libraries like Streamlit. Streamlit is compatible and integrates with many Python libraries including scikit-learn, Keras, PyTorch, NumPy and Pandas. Streamlit supports displaying text, data, interactive widgets, and many charting and graphing libraries including Matplotlib, Vega-Lite and Plotl (Dennis, 2022)

Streamlit is an open-source python library that is useful to create and share data web apps. It is slowly gaining a lot of momentum in the data science community. Because of the ease with which one can develop a data science web app, many developers use it in their daily workflow (Niggli, 2022). The Streamlit GitHub repository has more than 14.1k stars and 1.2k forks. Under the hood, it uses React as a frontend framework to render the data on the screen. So, React Developers can easily manipulate the UI with few changes in the code. Streamlit turns data scripts into shareable web apps. The figure 25 is a representation of coding transformation to the web app on the browser.

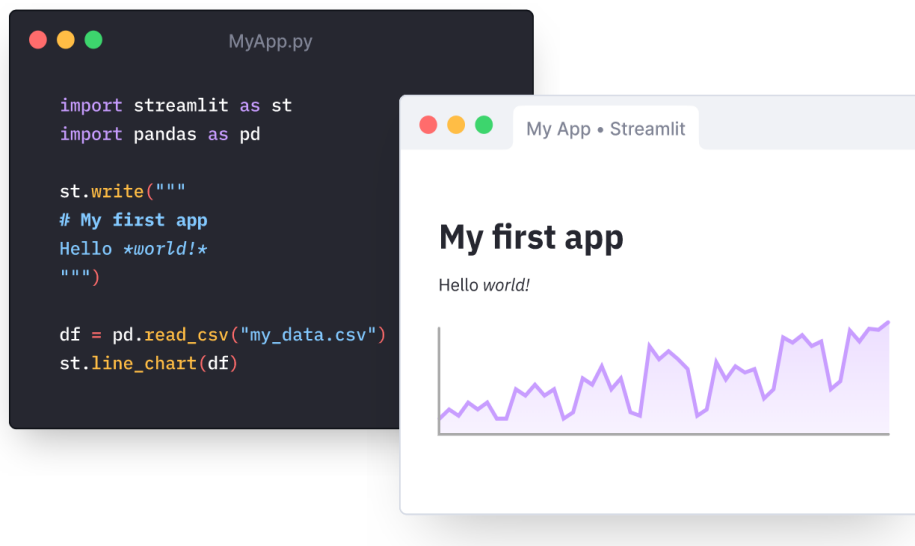


Figure 24: Streamlit code and view in browser

The way that Streamlit visualize the code and create a web app is totally innovated even if we are in the 2022. On the upcoming years we will see more and more libraries to be created with the similar score. The ML trend is on hype!

### 6.3. Implementation

The code for the implementation is not complicated. To run the application in browser, need to be very careful with the packets that you have installed. You must meet the following requirements in your local dev environment.

- Python version  $\geq 3.XXX$  (3.7 recommended)
- pip
- virtual environment
- Pycharm or any other Code Editor

In my case, I had to downgrade the existing python version and to upgrade some packages.

So, once the Streamlit is on air, we can view the on the web.

```
(basenew) C:\Users\efmak>streamlit run app2.py
You can now view your Streamlit app in your browser.
Local URL: http://localhost:8501
Network URL: http://192.168.2.2:8501
```

Figure 25: run the Streamlit app

The UX that we have created is simple as you may see in the below capture. On the left side you may see the filter panel, with which the user can pick the activity type sequence. On the right side, the table will be affected accordingly. The web app is a fully interactive. Of course, we added the option of export to csv file. Using this capability, the user can export the filtered customer group (table) and provide it to the CRM team to create a targeted campaign.

The app should be shareable. So, we deployed it via GitHub cloud service.



Your app is in the oven

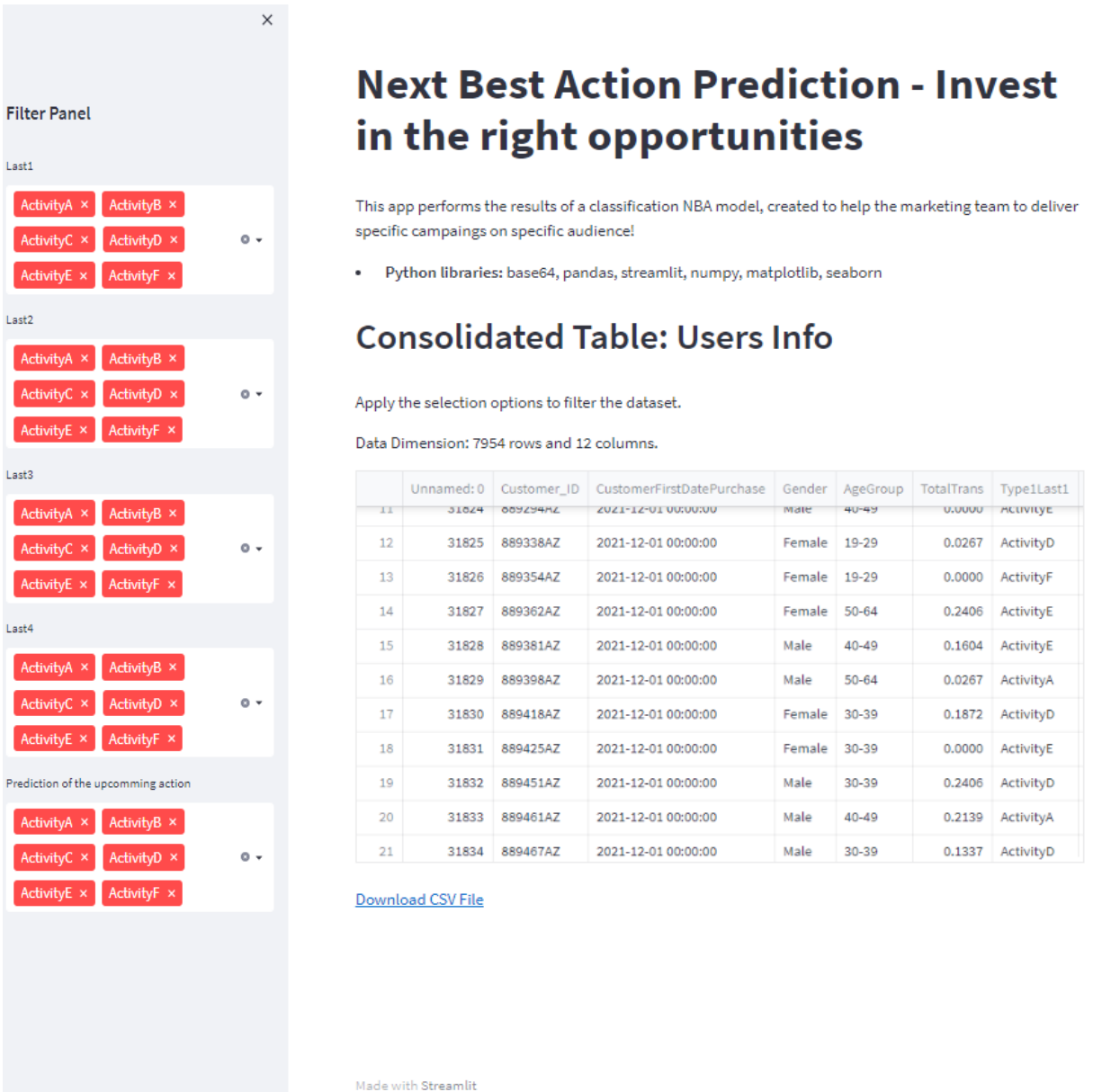
Figure 26: the Streamlit app is ready to deploy in cloud

Behind the scenes, Streamlit keeps track of the function name, the code in the function body, and the input arguments we pass in the function call. Upon first invocation, it stores the function's return value in a local cache. Upon subsequent invocations of the function with the exact same arguments, Streamlit returns the result from the local cache. The main limitation of Streamlit data caching is that it cannot keep track of changes to the data happening outside the function body (Treuille, 2022).



Visualization and Streamlit framework are not the core elements of this study, however the way of working and the aim to approach an end-to-end realistic solution, made me to dig deeper to new technologies and optimize the solution in all stages. So, we will not elaborate more about it at this thesis, while there is huge potential of usage and many functionalities to take advantage of it.

This is the [link](#) for the web app. Please explore! Below you may find how our Streamlit app looks like on your web browser.



The screenshot shows a web application interface with a 'Filter Panel' on the left and a main content area on the right. The 'Filter Panel' contains four sections labeled 'Last1', 'Last2', 'Last3', and 'Last4', each with six red buttons labeled 'ActivityA', 'ActivityB', 'ActivityC', 'ActivityD', 'ActivityE', and 'ActivityF'. Below the filters is a section titled 'Prediction of the upcoming action' with the same six buttons. The main content area features a large heading 'Next Best Action Prediction - Invest in the right opportunities', a descriptive paragraph, a list of Python libraries, a 'Consolidated Table: Users Info' section with a table of user data, and a 'Download CSV File' link. The footer of the page reads 'Made with Streamlit'.

## Next Best Action Prediction - Invest in the right opportunities

This app performs the results of a classification NBA model, created to help the marketing team to deliver specific campaigns on specific audience!

- Python libraries: base64, pandas, streamlit, numpy, matplotlib, seaborn

### Consolidated Table: Users Info

Apply the selection options to filter the dataset.

Data Dimension: 7954 rows and 12 columns.

	Unnamed: 0	Customer_ID	CustomerFirstDatePurchase	Gender	AgeGroup	TotalTrans	Type1Last1
11	31824	889234AZ	2021-12-01 00:00:00	Male	40-49	0.0000	ActivityE
12	31825	889338AZ	2021-12-01 00:00:00	Female	19-29	0.0267	ActivityD
13	31826	889354AZ	2021-12-01 00:00:00	Female	19-29	0.0000	ActivityF
14	31827	889362AZ	2021-12-01 00:00:00	Female	50-64	0.2406	ActivityE
15	31828	889381AZ	2021-12-01 00:00:00	Male	40-49	0.1604	ActivityE
16	31829	889398AZ	2021-12-01 00:00:00	Male	50-64	0.0267	ActivityA
17	31830	889418AZ	2021-12-01 00:00:00	Female	30-39	0.1872	ActivityD
18	31831	889425AZ	2021-12-01 00:00:00	Female	30-39	0.0000	ActivityE
19	31832	889451AZ	2021-12-01 00:00:00	Male	30-39	0.2406	ActivityD
20	31833	889461AZ	2021-12-01 00:00:00	Male	40-49	0.2139	ActivityA
21	31834	889467AZ	2021-12-01 00:00:00	Male	30-39	0.1337	ActivityD

[Download CSV File](#)

Made with Streamlit

Figure 27: This is the view of our custom Streamlit app on the web

The newly created web page will display the screen image that was shown above. This includes title, logo, filters panel, sidebar, a data table, download the selection in CSV. We have completed developing a web app using python and the Streamlit frameworks! We deploy our web application on the Streamlit cloud platform for free, supporting the open-source community and functionalities.

## 7. Conclusion

### 7.1. Outcome

The problem addressed in this thesis was to implement an attractiveness prediction - classification model and evaluate its performance. We tried to bring in research table a very crucial project for many companies, the prediction of customer next action around a service that the company provides. For our sequence-to-sequence problem, the LSTM worked with high correspondence reaching almost 60% of accuracy. On the churn experiment, LSTM worked even better, as it scored more than 80% in accuracy.

In this study, different LSTM-recurrent neural networks were tested. From the authors side, more insight is needed to further develop and assess the models and the certainty in these. From the literature review it seems that data driven model are a popular choice for customer analytics. Various machine learning tools can be utilized for such kind of modelling purposes. Those models can be a little tedious to build and understand for a novice, but when they are in place, they only require data. With all the new data that is created every day and the ongoing sensor-revolution LSTMs and other machine learning methods may be used for many interesting tasks and research projects within the customer experience world. Predicting the future is not easy when we are talking about customer behavior, but it is feasible. To predict tomorrow's value, feed into the model the past N activities and get the upcoming activity as output, that is really challenging in CX analytics. The dataset that used in each experiment had crucial effect to the results. It is very important to use the 'correct' data unless you want to drive into false insights and learnings. LSTM seems that worked better when the activity options are not so many and the volume of customers was around ~10K. Data has come along a long way in the past few years, from countable numbers to now sitting on countless data points. Data is generated at a faster pace than ever. But we can control the quality of data points, which will lead to the success of our AI models. Datasets are, after all, the core part of any Machine Learning project. Understanding and choosing the right dataset is fundamental for the success of an AI project.

Looking from a different angle, in this thesis the focus was not only to the technical part, but there was also a lot of business logic throughout the data collection, the implementation and the interpretation of the results. Data analysts/scientists have to conquer the business needs in order to have value and impact to any organization. This thesis teaches me how complex is the from scratch process, and the most important thing is that every output must be conform with the business requirements and strategy. Data-driven companies treat their data as a business asset and actively look for ways to turn it into a competitive advantage. Success with business analytics depends on data quality, skilled analysis, the tools -technologies and the business understanding. Models, as those in the current thesis, help enterprises make more informed, real decisions with business processes. With the aim of working on a realistic approach, the above need made the work more challenging.

Of course, the fact that I had to work with all those tools, program languages, modern techniques made me more motivated to continue my journey in machine learning/deep learning applications. Moving to a data-driven business model – where decisions are

made based on what we know to be true rather than “gut feeling” – is core to the wave of digital transformation sweeping through every industry in 2023 and beyond. It helps us to react with certainty in the face of uncertainty – especially when wars and pandemics upset the established order of things. But the world of data and analytics never stands still. New technologies are constantly emerging that offer faster and more accurate access to insights. And new trends emerge, bringing us new thinking on the best ways to put it to work across business and society at large. So, the way of working here, the issues that I had to face and the optimization that I had to make in order to achieve better results, were built on the most up to dated tools and techniques, made me feel like a lifelong learner and full of hungriness for more education and practice.

Overall, this work provides promising initial work in prediction customer behavior using the past activities that he/she made so that the companies could use personalized marketing and communication techniques to target specific audience and to avoid churn. This work opens up many avenues for future work, such as fine tuning the neural network designs and their hyper parameters, changing the design of the layers and/or combine different types of RNNs to further refine the results. Selecting customer behavior parameters also had a great influence on the results. This work could be extended by further investigating the cx parameters and their contributions to the prediction process. This could be achieved by either statistical means or going deeper in the analytical and empirical theories and equations to provide a deeper understanding of the relations between parameters, and thus, more precise future predictions. I strongly believe that this work will encourage other researchers to try LSTM for sequence-to-sequence related work in any kind of experiments.

## **7.2. Future Work**

There are no limitations on customer behavior analytics. In an ideal data world without GDPR restrictions, we can bring to life any data initiative. So below some of the thoughts that could be very interesting to implement in the future with similar kind of data and purposes.

Assume that you have some customers digital footprints (activities) for a specific time. These customers having many months/years as customer lifetime but for some reason you have not track their early first activities, you missed the begging of the activity sequence. What if you could work backwards to investigate what did they do in the past as first steps to reach the segment that they belong now?

Of course, this kind of predictions could be combined with an NLP model to generate full personalized communication messages for the customer, to make him/her feel individual. As we mentioned above, today’s customer wants to feel unique, moreover the long-time loyal customers.

The automation of the execution could not be missed from this future work list. As we used the Streamlit library of python to visualize the application, with the aim to create a self-service and easy way of segments filtering for the stakeholders, it would be a great idea to scale up this web app by generate a new prediction with it.

In real world, customer behavior is affected by a lot of external factors. Seasonality,

the discounts, the weathers, the economy, the location, the gender, the birthday, even the mood are playing a very crucial role on customer choices. To be more accurate with the real world, importing those factors into the model, It would a step for more realistic solutions. Therefore, the model's performance may be better if the models are dynamically updated through time. To take the performance into a higher level, there need to be considered features that involve information for the companies' characteristics about social and financial state.

As I am an experienced user of R programming language, I would be very interesting to try the above data deep learning solutions with R. It is a fact that python is the most preferable coding language now when talking about machine learning and deep learning but there are other ways of working with continuous improvements through the years that may serve similar or even better results.

Finally, a valuable explore could be to focus on patterns of sequence that presents an anomaly behavior on customer, like what makes him/her to make a big amount purchase, or what make him/her to be an offline user (user without interaction) for such a long time? This pattern sequence recognition could give solution to many business questions.

Although, considering that nowadays we have a strong tool that models nonlinear relationships, Neural Networks, we should be more than unstoppable on researching. There is plenty of room for future work on customer analytics stage and many things to learn yet. It is a challenging sector for the researchers as it is characterized by many as a chaotic system.

## Bibliography

- Bazylevska, O. V. (2021). *Customer analytics as a source of competitive advantage*.
- Bediako, P. K. (2017). *Long Short-Term Memory Recurrent Neural Network for detecting DDoS flooding attacks within TensorFlow Implementation framework*.
- Bourdakos, N. (2018). Retrieved from Medium: <https://medium.com/hackernoon/train-your-machine-learning-models-on-googles-gpus-for-free-forever-a41bd309d6ad>
- Brownlee. (2014). *Long Short-Term Memory Networks With Python*.
- Brownlee, J. (2016). *Time Series Prediction with LSTM Recurrent Neural Networks in Python with Keras*. Retrieved from Machine Learning Mastery: <https://machinelearningmastery.com/time-series-prediction-lstm-recurrent-neural-networks-python-keras/>
- Chng, Z. M. (2022). *Google Colab for Machine Learning Projects*. Retrieved from Machine Learning Mastery: <https://machinelearningmastery.com/google-colab-for-machine-learning-projects/>
- Chong, D. (2020). *Deep Dive into Netflix's Recommender System*. Retrieved from <https://towardsdatascience.com/deep-dive-into-netflixs-recommender-system-341806ae3b48>
- Choubey, V. (2020). *Understanding Recurrent Neural Network (RNN) and Long Short Term Memory(LSTM)*. Retrieved from <https://medium.com/analytics-vidhya/undestanding-recurrent-neural-network-rnn-and-long-short-term-memory-lstm-30bc1221e80d>
- Dago Diedrich, N. N.-S. (2021). Retrieved from Mckinsey: <https://www.mckinsey.com/capabilities/strategy-and-corporate-finance/our-insights/strategic-resilience-during-the-covid-19-crisis>
- Dennis, N. (2022). Retrieved from Creating an Awesome Web App With Python and Streamlit: <https://python.plainenglish.io/creating-an-awesome-web-app-with-python-and-streamlit-728fe100cf7>
- Dey, V. (2021). *When to Use One-Hot Encoding in Deep Learning?* Retrieved from DEVELOPERS CORNER: <https://analyticsindiamag.com/when-to-use-one-hot-encoding-in-deep-learning/>
- Dietterich, T. G. (2004). *Machine Learning for Sequential Data: A Review*. PGP Public Key.
- Dires Negash Fente, D. K. (2018). *Weather Forecasting Using Artificial Neural Network*.
- Dunkley, C. (2022). *Churn Prediction: The Basics of Predicting & Preventing Churn*. Retrieved from <https://mode.com/blog/predicting-and-preventing-churn/#:~:text=Churn%20prediction%20is%20predicting%20which,their%20behavior%20with%20your%20product.>
- Duran, O. D. (2020). *Studying the Benefit of Classical and Deep-Learning Predictors on Ridesharing Optimisation*.
- Dykes, B. (2021). Retrieved from Deloitte: <https://www.forbes.com/sites/brentdykes/2021/06/01/10-reasons-why-your-organization-still-isnt-data-driven/?sh=7006d13b7d80>
- Fabio Carrara, P. E. (2019). *LSTM-Based Real-Time Action Detection*. Springer.
- Farshad Kooti, L. M. (2016). *Portrait of an Online Shopper: Understanding and Predicting Consumer Behavior*. San Fransisco, USA.
- Fontanella, C. (2022). Retrieved from Hubspot: <https://blog.hubspot.com/service/customer-behavior-analysis>
- Grace. (2021). *How to Use Customer Analytics the Right Way (and Why It Matters)*. Dialpad.
- Gulli, A. (2019). *Deep Learning with Tensorflow 2 and Keras*. Packt Publishing Ltd.
- hemavatisabu. (2018). *GeeksforGeeks*. Retrieved from <https://www.geeksforgeeks.org/data-pre-processing-wit-sklearn-using-standard-and-minmax-scaler/>
- Horowitz, A. (2014). Retrieved from a16z: <https://a16z.com/ai-survey/dl-architectures/>
- Inf, X. (2020). *Pre-padding and post-padding in LSTM*. Retrieved from <https://medium.com/@xin.inf19/pre-padding-and-post-padding-in-lstm-975faff4efa5>
- Irshad, T. (2020). *Long Short-Term Memory Decoded*. Retrieved from

- <https://medium.com/analytics-vidhya/long-short-term-memory-decoded-9041fe06235f>
- Jane Priest, S. C. (2013). *Consumer Behavior*. Edinburgh Business School: Heriot-Watt University.
- Jason, B. (2017). *Gentle Introduction to the Adam Optimization Algorithm for Deep Learning*. Retrieved from <https://machinelearningmastery.com/adam-optimization-algorithm-for-deep-learning/>
- Jayawardhana, S. (2020). *Sequence Models & Recurrent Neural Networks (RNNs)*.
- Jeon, G. (2019). *A Novel Time Series based Seq2Seq Model for Temperature Prediction in Firing Furnace Process*.
- John, B. (2022). *How to Implement Customer Churn Prediction [Machine Learning Guide for Programmers]*. Retrieved from <https://neptune.ai/blog/how-to-implement-customer-churn-prediction>
- Karpathy, A. (2015). *The Unreasonable Effectiveness of Recurrent Neural Networks*.
- Keith, M. (2021). *Exploring the LSTM Neural Network Model for Time Series*. Retrieved from <https://towardsdatascience.com/exploring-the-lstm-neural-network-model-for-time-series-8b7685aa8cf>
- Kenton, W. (2021). *What Is Attribute Sampling?* Retrieved from <https://www.investopedia.com/terms/a/attribute-sampling.asp#:~:text=What%20is%20Attribute%20Sampling%3F,controls%20are%20being%20correctly%20followed.>
- Khan, R. (2022). Retrieved from Importance of Datasets in Machine Learning and AI Research: <https://www.datatobiz.com/blog/datasets-in-machine-learning/>
- Kostanje, J. (2021). *Advanced Forecasting with Python*. Apress.
- Kösters, M. (2020). *The Five Benefits of Data Visualization*. Retrieved from <https://www2.deloitte.com/nl/nl/pages/tax/articles/bps-the-five-benefits-of-data-visualization.html>
- Lai, G. (2021). *Neural Sequential Modeling and Applications*.
- LENDAVE, V. (2021). *A Tutorial on Sequential Machine Learning*.
- M. Goutham, S. S. (2021). *User Activity Sequence Prediction in Smart Homes using Multi-Layer Long Short-Term Memory Networks*. Deaborn: Ford Motor Company.
- Mjaaland, H. L. (2020). *Detecting Fake News and Rumors in Twitter Using Deep Neural Networks*.
- Morgan, B. (2022). Retrieved from Forbes: <https://www.forbes.com/sites/blakemorgan/2022/05/01/the-top-100-most-customer-centric-companies-of-2022/?sh=7ad750502b38>
- Murariu, C. (n.d.). *How to make the most of your CRM using customer behavior data*. Retrieved from innertrends: <https://www.innertrends.com/blog/make-crm-using-customer-behavior-data>
- Musumano. (2019, May). *sevenrooms*. Retrieved from What Is Customer Segmentation? Here's The Definition.
- Niggel, D. (2022). *Creating an Awesome Web App With Python and Streamlit*. Retrieved from <https://python.plainenglish.io/creating-an-awesome-web-app-with-python-and-streamlit-728fe100cf7>
- Osborne, C. (2021). Retrieved from Timelinegenius: <https://blog.timelinegenius.com/how-to-use-analytics-to-optimize-your-messaging/>
- Pai, A. (2022). *What is Tokenization in NLP? Here's All You Need To Know*. Retrieved from <https://www.analyticsvidhya.com/blog/2020/05/what-is-tokenization-nlp/>
- Patel, A. (2022). Retrieved from Forbes: <https://www.forbes.com/sites/forbestechcouncil/2022/11/07/top-five-data-science-trends-that-made-an-impact-in-2022/?sh=2c3f158f57f4>
- PeterCLu. (2021). Retrieved from Microsoft: <https://learn.microsoft.com/en-us/azure/machine-learning/component-reference/normalize-data>

- PeterCLU. (2021). *Partition and Sample component*. Retrieved from <https://learn.microsoft.com/en-us/azure/machine-learning/component-reference/partition-and-sample>
- Pilladin, S. R. (2021). *Applying AI and Machine Learning to Predict Consumer Behavior*. Retrieved from section.io: <https://www.section.io/engineering-education/applying-ai-and-ml-to-predict-consumer-behavior/>
- Pykes, K. (2022). *What is Data Visualization? A Guide for Data Scientists*. Retrieved from <https://www.datacamp.com/blog/what-is-data-visualization-a-guide-for-data-scientists>
- Radu, V. (2022). Retrieved from Omniconvert: <https://www.omniconvert.com/blog/consumer-behavior-in-marketing-patterns-types-segmentation/>
- ResearchDive. (2022). <https://www.researchdive.com/covid-19-insights/194/customer-analytics-market>. Retrieved from <https://www.researchdive.com>.
- Rithika, S. (2022). Retrieved from Building a Churn Prediction Model on Retail Data Simplified: The Ultimate Guide 101: <https://hevodata.com/learn/churn-prediction-model/>
- Robail Yasrab, M. P. (2020). *PhenomNet: Bridging Phenotype-Genotype Gap: A CNN-LSTM Based Automatic Plant Root Anatomization System*.
- Robail Yasrab, M. P. (2020). *PhenomNet: Bridging Phenotype-Genotype Gap: A CNN-LSTM Based Automatic Plant Root Anatomization System*.
- Rohn, S. (2022). *What Is Customer Analytics? +Benefits, Types, Best Practices*. Retrieved from whatfix: <https://whatfix.com/blog/customer-analytics/>
- Roman, V. (2020). *Recurrent Neural Networks!* Retrieved from Towards Data Science: <https://towardsdatascience.com/recurrent-neural-networks-56e1ad215339>
- Sengupta, A. (2022). Retrieved from tutorialspoint: <https://www.tutorialspoint.com/what-are-the-factors-affecting-consumer-behaviour>
- Shanmugamani, R. (2018). *Deep Learning for Computer Vision*. Packt Publishing.
- Shirk, C. (2021). Retrieved from Crm: <https://crm.org/crmland/analytical-crm>
- Shrott, R. (2019). Retrieved from Sequence Models by Andrew Ng — 11 Lessons Learned: <https://towardsdatascience.com/sequence-models-by-andrew-ng-11-lessons-learned-c62fb1d3485b>
- Simplilearn. (2022). *What Is Keras: The Best Introductory Guide To Keras*. Retrieved from <https://www.simplilearn.com/tutorials/deep-learning-tutorial/what-is-keras>
- Sindle, J. (n.d.). Retrieved from How Machine Learning Is Improving Real-Time Customer Experience: <https://www.alterian.com/articles/how-machine-learning-is-improving-real-time-customer-experience/>
- Singh, A. (2020). *All You Need To Know About RNN*. Retrieved from <https://www.kaggle.com/questions-and-answers/199804>
- snehankekre. (2022). Retrieved from Streamlit: <https://github.com/streamlit/streamlit>
- Starikova, S. (2018). *IMPLEMENTING A/B TEST RESULTS ON THE E-COMMERCE WEBSITE*. Oulu University of Applied Sciences.
- TensorFlow. (2022). *TensorFlow 2*. Retrieved from TensorFlow: <https://www.tensorflow.org/tutorials/quickstart/beginner>
- Tkachova, Y. (n.d.). Retrieved from Masthead: <https://mastheadata.com/how-important-data-quality-for-machine-learning/#:~:text=The%20data%20used%20in%20machine,of%20quality%20of%20its%20data.>
- Tozzi, C. (2021). Retrieved from TechTarget: <https://www.techtarget.com/searchcloudcomputing/tip/Open-source-cloud-platforms-and-tools-to-consider>
- Treuille, A. (2022). *Creating an Awesome Web App With Python and Streamlit*. Retrieved from Medium: <https://medium.com/streamlit/introducing-streamlit-components-d73f2092ae30>
- Williams, D. (2019). *Campaign Strategy: 3 Steps to Improve Messaging Using Data*.



Yasar, K. (2021). Retrieved from techtarget:

<https://www.techtarget.com/searchcustomerexperience/definition/CRM-analytics>

Zadransk´a, L. (2019). *Time Series Forecasting*.