DEGREE PROJECT IN MEDICAL ENGINEERING,
SECOND CYCLE, 30 CREDITS
STOCKHOLM, SWEDEN 2019

Predicting Patient Length Of Stay at Time of Admission Using Machine Learning

OLLE ANDERSSON

KTH ROYAL INSTITUTE OF TECHNOLOGY
SCHOOL OF ENGINEERING SCIENCES IN CHEMISTRY, BIOTECHNOLOGY AND HEALTH

Abstract This master thesis investigates the possibility of using machine learning methods to predict patient length of stay at the time of admission to a clinical ward from the emergency department. The main aim of this thesis is to provide a comparative analysis of different algorithms and to suggest a suitable model that can be used in a hospital prediction software. The results show that it is possible to achieve a balanced accuracy of 0.72 at the time of admission and of 0.75 at a later stage in the process. The suggested algorithm was Random Forest which combines good accuracy with effective training time, making it suitable for on-line use in a hospital. The study shows that there is a clear potential for the use of machine learning methods for predicting length of stay, but that further improvements have to be made before adaption into the healthcare.

Sammanfattning Detta masterexamensarbete utforskar möjligheten att använda maskininlärning för att förutspå vårdtiden för en patient då denne skrivs in på en vårdavdelning från akutvårds-avdelningen vid ett sjukhus. Huvudmålet för arbetet är att tillhandahålla en jämförelse av olika maskininlärnings-algoritmer och föreslå en algoritm som är lämplig att integrera i en mjukvara på sjukhuset. Resultaten visar att det är möjligt att nå en balanced accuracy på 0.72 vid inskrivningstillfället samt 0.75 vid en senare tidpunkt i vårdprocessen. Den föreslagna algoritmen var Random Forest som kombinerade bra prestanda med effektiv träningstid, något som gör den lämplig för att köras på sjukhuset. Projektet visar att det finns en tydlig potential för att använda maskininlärning för att prediktera vårdtid men att förbättringar krävs innan det kan nå hela vägen in i sjukhuset.

Contents

1 Acknowledgements

First of all, I would like to thank my KTH supervisor Pawel Herman for valuable input and feedback as well as Zenodia Charpy who with great excitement and ambition also has helped to push this project in the right direction. I would also like to thank everyone at Cambio for being excited and supportive in this endeavour and providing a wonderful atmosphere to work in.
One of the main supporters of this project has been Christian Kylander, who has taken valuable time from his very busy schedule to give insights and thoughts on the experiments. In fact, the project would probably not have been possible without his support. I hope he will continue to believe and be interested in machine learning in the future and help take this project all the way.
Finally, and most importantly, I would like to extend a huge thank you to my supervisor at Cambio, Marcus Petersson. Without his support and engagement in this project, it would never have gotten as far as it did and I am grateful that I got the chance to work with it. I hope that this project can, at least, act as a stepping stone in the right direction and that he continues to strive toward bringing ML-based solutions into the healthcare with the same passion as he showed for this project.

2 Introduction

The healthcare sector is facing ever increasing challenges. Increased life expectancy means the percentage of retired, elderly people is predicted to increase significantly the coming years. In a forecast of the Swedish population growth published by SCB, an increase of roughly 200 000 individuals in the age group 80+ is expected between 2020 and 2030 [1]. Even though other age groups are predicted to grow slightly as well, the result is that the working population needs to support an increasingly larger elderly population. Combined with economical challenges, lack of expertise, staff and hospital beds, it is clear that the healthcare will have to battle enormous challenges in the years to come.
In order to face these challenges, hospitals need to be equipped with administrative planning tools that can allow them to allocate the available resources in an efficient manner. Many different methods have been implemented and explored for these purposes. Hospitals have tried to streamline and standardize clinical procedures and patient flows in order to allow a higher throughput of patients. Bottlenecks have been identified and restructured and new ways of patient interaction have been tested by using for example mobile apps.
There is, however, a new area of interest that could bring new tools to the hospitals' administrative toolbox, namely machine learning (ML). Machine learning has been on the rise in several fields for the last decade following improved computational power, availability of data and improved algorithms. It has excelled in tasks such as image segmentation and classification, machine translation and recommender systems. In medicine, machine learning has shown promise in segmenting and classifying radiology images [2], diagnosing and identifying high-risk patients [3] and the topic of this project: predicting length of stay (LoS).
Patient length of stay is most commonly defined as the total hospitalization time, i.e. from admission to discharge. LoS predictions can be used in many different ways and serve as a very valuable method for resource planning. Not only could it provide an overview of future bed capacity, but it could also be used as a precautionary warning that extra measures should be taken given that a patient's LoS might be longer than usual, such as social planning or extra medical attention. Additionally, certain ML methods can potentially provide valuable insights into what features affect LoS and hence be used as a way to evaluate procedures in order to more efficiently treat patients and reduce unnecessary workload.
The aim of this thesis is to provide a comparative analysis of different, commonly used ML-algorithms applied to the LoS prediction task and to answer if it is actually possible to achieve sufficiently good predictive performance. Additionally, different aspects of applying ML methods in the actual healthcare setting are discussed and used to evaluate the algorithms with the final aim of suggesting a suitable model for the task.
Furthermore, a key aspect of this project is that the dataset covers a wide range of patient types in terms of diseases, something that is not commonly used in previous LoS research. Typically, the dataset is limited to a certain disease or patient type. By exploring the possibilities of predicting LoS for a wide variety of patients, this project contributes to the challenge of applying ML methods at a more generalized level in the healthcare process which is an important step in the process of bringing these types of solutions into the healthcare setting.

3 Method

3.1 Dataset

The dataset used in this study was collected from a Swedish hospital over a two-year period between 2017 and 2019. More specifically, the dataset consisted of patients admitted to a specific ward of the hospital from the emergency room (ER).
Collected parameters included vital parameters, chemistry tests, priority marking, radiology procedures, suspected diagnosis, planned surgeries, previous medical history along with general information such as age, sex and marital status.
In order to protect patient integrity, the dataset was available only at the hospital's servers and all patient-specific data was encoded using UUID-codes. Using this method, patient privacy remained intact while still allowing for systematic analysis of the different codes, data pre-processing and application of machine learning algorithms. Using only categorical and numerical data also meant that the structure of the data was easier to handle as input to the algorithms compared to for example including free-text journal data which is harder to process and might risk exposing personal and vital patient information.

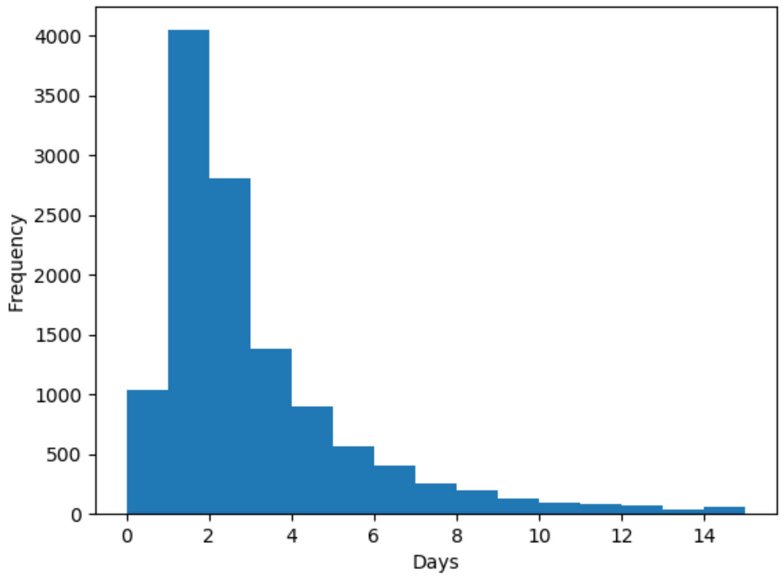
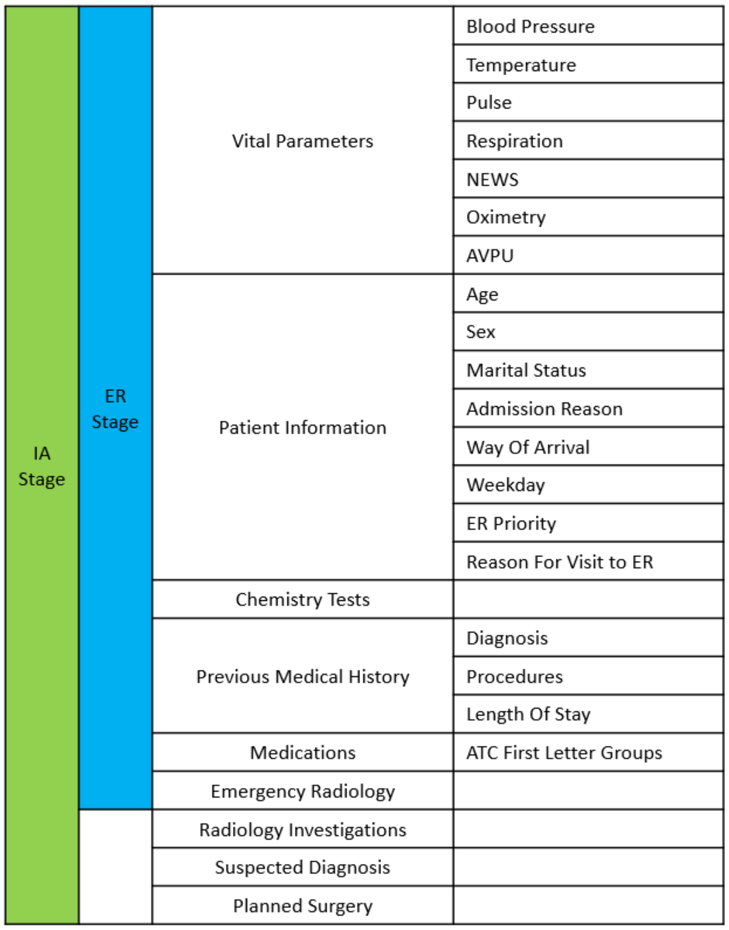


Figure 1: Distribution of LoS for patients arriving through the ER. Excluding patients with a LoS longer of 15 days.

3.2 Pre-processing

This project chose to focus only on those patients that arrived through the ER and then went on to be admitted to the clinic instead of also including elective patients which were available in the base data. The first reason was that from a clinical viewpoint, patients arriving from the ER were of higher interest due to their unscheduled nature and varying characteristics.
The second reason was that the data collection process in the ER was standardized which meant that missing values were uncommon. This resulted in that patients that lacked values in one of the vital parameters could be dropped from the dataset completely instead of imputed without losing a large amount of training samples. The reason for not imputing vital parameter measurements is



IA Stage	ER Stage	Vital Parameters	Blood Pressure
			Temperature
			Pulse
			Respiration
			NEWS
			Oximetry
			AVPU
		Patient Information	Age
			Sex
			Marital Status
			Admission Reason
			Way Of Arrival
			Weekday
			ER Priority
			Reason For Visit to ER
		Chemistry Tests	
		Previous Medical History	Diagnosis
			Procedures
			Length Of Stay
		Medications	ATC First Letter Groups
		Emergency Radiology	
		Radiology Investigations	
		Suspected Diagnosis	
		Planned Surgery	

Figure 2: The feature set used in this project. The overlapping IA Stage means that at this stage data collected from both the ER and the ward is included in the dataset (as separate feature groups).

discussed further in section 3.3.2.
Finally, patients with a LoS longer than 15 days were removed because they were considered outliers. These patients occurred very rarely and while still being clinically important, their circumstances were usually too unique to be included in any sort of pattern. The final dataset consisted of 12076 contacts. The word contact is used here instead of patient because the same patient could visit the hospital twice and each entry is therefore not regarded as a unique patient, but rather a single contact with the hospital.

3.3 Feature Engineering

The dataset included a lot of time-series data as well as categorical data in form of diagnoses, prescriptions and radiology investigations. This meant that an extensive part of this project was put into building a feature extraction pipeline in which manual features were extracted from the time-series data. In the following sections, specific details about extracted features are explained.

3.3.1 Vital Parameters

One of the most abundant data in this dataset consisted of vital parameters measurements both from the ER department and the clinic. The vital parameters included measurements of blood pressure, temperature, pulse, respiration, oximetry and BMI. Furthermore, an alertness ranking called AVPU (Alert, Voice, Pain, Unresponsive) was included as well as the patient's NEWS (National Early Warning Score). The vital parameters therefore consisted of both numerical and categorical data.
For the time-series data, a standard set of features were extracted for each specific parameter. These included start- and end values, mean, variance and the number of measurements made. Start- and end values were extracted in order to give a notion of the trend. For example, an admitted patient could have abnormal start blood pressure values but after treatment with medicine at the ER these values could stabilize and so the end value would be normal.
Variance was extracted in order to represent how stable a vital parameter was during the stay. A high variance in for example pulse or respiration could indicate that the patient has some serious underlying issue but a high variance could also indicate that the patient has recovered from earlier abnormal values.
For certain parameters such as oximetry, special indicators were included. If the patient has a oxygen saturation lower than 90 % at any time during the ER or ward stay, a binary indicator feature for hypoxia was set to one. This way, a higher resolution of features could be presented to the algorithm even if it meant that redundant features were introduced into the feature set.

3.3.2 Chemistry Tests

The other large feature group was chemistry tests. In contrast to vital parameters, there were several hundred different chemistry tests available in the data and in order to keep the number of features down only the top 15 most common chemistry tests were included. These tests also had enough support such that not more than 50 % of contacts needed to have a value imputed.
In comparison to vital parameters, chemistry tests were considered to be of a less dynamic nature and hence more reasonable to impute. While vital parameters can change over the course of hours or even minutes with the use of for example medication, chemistry values change less rapidly and can be stable over several days or weeks. Taking this into consideration along with the fact that vital parameter trends are one of the most important considerations clinically [4], it was decided to impute chemistry values while dropping patients with missing vital parameter measurements.

3.3.3 Diagnoses, Medication And Other Categoricals

Other information both collected during and before the patient stay consisted of diagnoses, medication, procedures and radiology investigations. This data consisted of many unique codes for each specific medicine or diagnosis. Most codes were uncommon with a few 100 contacts each and including every code would have resulted in a very sparse feature set and therefore it was important to filter out the most relevant data. A too high-dimensional dataset would mean that the risk of including noise increases and could have a negative impact on the classification accuracy.
For diagnoses and procedures, the top 100 most common codes were included as separate, binary features as the support for these were substantial enough to provide additional information compared to diagnosis codes outside the top 100. For medication, the ATC (Anatomic Therapeutic Chemical) coding system was used to avoid including too many features. ATC is a hierarchical coding system with 14 main groups that shows where or how the medicine is working in the body. The first letter in the ATC coding was extracted (and encoded to keep patient privacy protected) and used as a feature. This meant that medication could be represented using 11 features (out of the 14 main groups) compared to the several thousands of different medication codes available if

a lower level of the ATC system was chosen. Only medication received in the last year from the contact start time was included.
Radiology was treated similarly to diagnoses by extracting the top 25 most common investigations. In addition, the number of different diagnoses, medications, procedures and radiology investigations each patient had undergone was also extracted as separate features.
Other categorical features included in this data set were age group, weekday admitted, sex, marital status and emergency marking/priority and these were encoded using one-hot encoding. For marital status, emergency marking and priority, missing values were filled in by the most common value. Table 2 shows the feature set included.

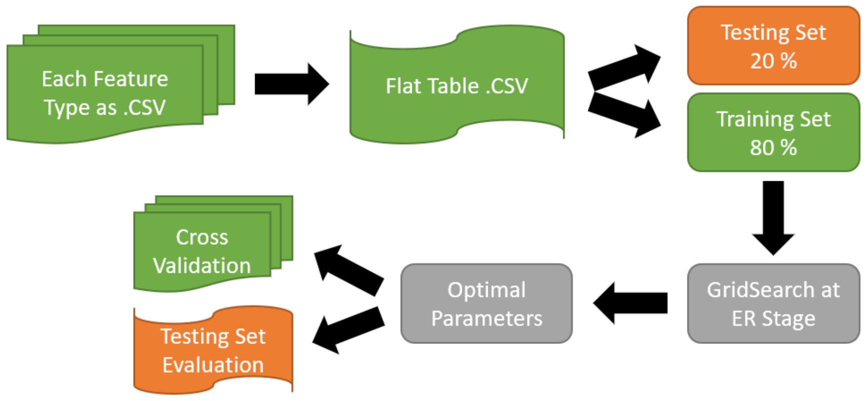


Figure 3: Overview of the workflow used in this project starting with separate, raw .CSV-files for each data type, construction of a flat table with one row per patient contact and then the process of obtaining optimal hyperparameter settings.

3.4 Problem Formulation

3.4.1 Classification Structure

In consultation with a physician from the clinic it was decided to tackle the prediction problem as a classification task with two classes. There were three main reasons for this. First of all, it was decided that regression would give more information than necessary. From a clinical perspective it could result in doubt and mistrust in the system from the healthcare staff if the given prediction was too precise meaning that a potential adoption of a prediction system would be more difficult.
The second reason was that it was necessary to simplify the task as much as possible. It would be better to start from the simplest scenario of a binary classification task and in line with the project's research aim ask if it was simply possible to do such a prediction given the available data. This approach was also similar to how earlier studies had approached the problem.
The last reason was that the information provided by a binary task would be enough for the clinic to get an indication whether or not the patient would be in the risk zone of deviating from the mean LoS. This would then give the clinicians a hint that the algorithm recognizes something potentially problematic with the patient, requiring further investigation and action from the staff.
On the basis of this, the final task was set as a binary classification with a split between the classes of three days. The mean LoS of the data was 2.59 days and a split of three days would therefore give an indication whether or not the patient would be above or below the mean.

This split was also necessary due to class imbalance. As can be seen in fig. 1, the number of long-staying contacts drastically decreases after the mean LoS. This meant that splitting at around the mean would result in good balance between the two classes. Even though the split was set to slightly longer than the mean, the resulting imbalance was still large which meant that splitting even further away from the mean would yield an almost empty long class. The class imbalance can be seen in fig. 4.

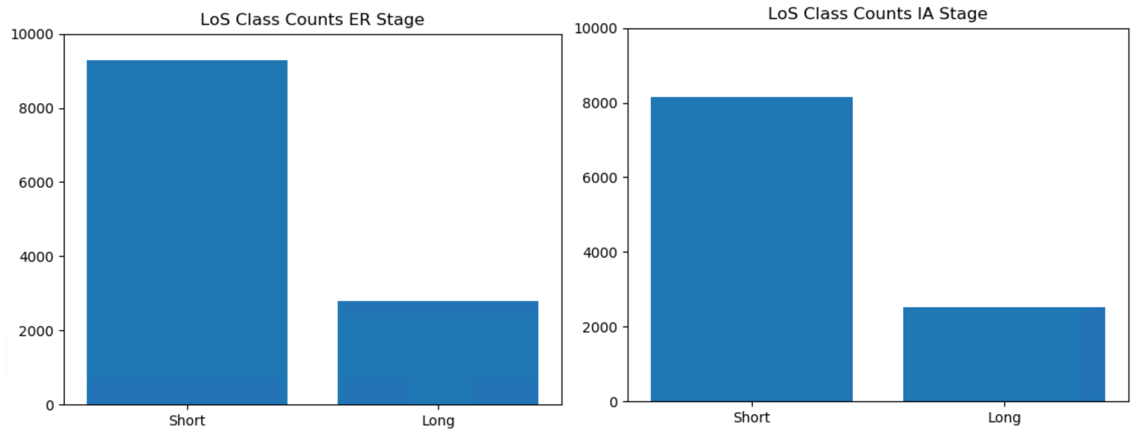


Figure 4: The counts for each class at both stages. The heavy imbalance between the two classes is a problem for many ML algorithms but can be combatted with different sampling schemes.

3.4.2 ER vs. In-Admission Stage

In order to utilize all available data, two different approaches were defined for when the prediction would be made. The first approach, in this project called the Emergency Room (ER) Stage, consisted of all the data collected from the ER up until the moment the patient was admitted to the ward. Complemented by the patients' earlier medical history, this data provided a good basis for an early prediction and warning when the patient arrive to the clinic. The dataset for this stage consisted of 12076 contacts and 380 features.

The second stage was named the In-Admission Stage. The idea behind this approach was that while it would be of high value for the clinic to receive the prediction at admission time, it could be possible to wait a certain amount of time if the prediction given at a later time could be more precise. The reason was that a lot of new information was collected about the patient at the ward. For example, a suspected diagnosis and planned surgery was set quite early in the process, resulting in more patient-specific information being made available the further into the admission the prediction is made.

The main information complementing the the ER stage was more vital and chemistry tests, more radiology investigations and added surgery information. When setting the prediction time to 12 hours into the admission, the total number of contacts was reduced to 10674 and the total number of features was increased to about 500. The contacts missing between the ER and In-Admission stage mostly lacked certain vital parameter measurements and was therefore, in line with the imputation strategy explained above, dropped.

3.4.3 Evaluation

In order to evaluate each algorithm, several evaluation criteria were used. These criteria were identified as interesting from both a machine learning and healthcare perspective, as the aim of evaluation was that the algorithm should both perform well but also be suitable to adopt at the clinic.
To evaluate the performance, the main metric was set to balanced accuracy. Balanced accuracy is defined as the average of the obtained recall for each class. Balanced accuracy reflects the accuracy better when the class imbalance is high, which is the case for the dataset used in this study. In addition to balanced accuracy, precision and recall were also used. The definitions of these metrics are presented in section A.5.3.
From the hospital perspective, the training time, prediction time and interpretability were also used as evaluation criteria when discussing the algorithms.

3.5 Machine Learning Algorithms

As the aim of this thesis was to investigate and compare different machine learning algorithms performance on the task of predicting LoS, a set of models had to be chosen to investigate. Mainly, algorithms that had proven to perform well in earlier LoS research were selected, as it would be interesting to see how these types of algorithms could handle a broader range of patients. A big consideration in picking the algorithm was that it should provide a high degree of interpretability. In this project, interpretability was said to constitute two things. First of all, an algorithm can be interpretable in the sense that a broader audience can understand how the algorithm works in terms of optimization and prediction. In the healthcare sector, this is an important factor because medical staff has to know the basis of every medical decision. Secondly, interpretability can be to what degree the algorithm provide tools and possibilities to visualize what features are important for the model, for example feature importance. This could be important when incorporating the model into software.
The final algorithms chosen to be investigated in this project was Decision Trees (DT), Random Forest (RF), Gradient Boosted Trees (GB), Support Vector Machine (SVM), AdaBoosted DT, AdaBoosted RF and finally neural network in the form of a Multi-Layer Perceptron (MLP). Both DT and RF provide a very high degree of interpretability and has been proved to perform well on the task. Gradient Boosting and AdaBoost were picked to see if boosting could improve the performance of the base models as this has been a common method to improve accuracy using weak learners. Finally, SVM and MLP were picked partly due to earlier performance in research, but also because they are generally strong algorithms that have been applied successfully for other tasks. More details on the specifics of each algorithm are presented in section A.6.

3.6 Training, Testing And Subsampling

The two datasets were split with 80 % as training data and 20 % as testing data. As mentioned before, there was a heavy imbalance between the two classes which caused problems when training the algorithms. Commonly, this results in a prediction bias towards the majority class. Many different resampling strategies are available such as random oversampling, random undersampling and more sophisticated techniques such as SMOTE. The main strategy used in this project was random undersampling in which the majority class is undersampled down to the same size as the minority class. This meant that the effective training size was reduced to 4482 contacts at the ER stage and 4038 contacts at the IA stage. The final number of contacts included in the training set are shown in table 1.

	ER stage	IA stage
Short class contacts	2241	2019
Long class contacts	2241	2019
Total contacts	4482	4038

Table 1: The number of contacts in each class, at each stage, after the undersampling step which made up the training data in all experiments except the subsampling experiment.

3.7 Grid Search

In order to achieve the best performance it was necessary to find the best hyper-parameters for each algorithm. A grid-search was conducted to find the best parameters for each model at the ER stage, which was then also used at the IA stage. A grid-search combines all possible combinations in a parameter grid where one defines the possible values for each hyper-parameter. In other words, it provides an exhaustive method to evaluate combinations of hyper-parameters.
To evaluate the model performance, grid-search uses cross-validation combined with scoring parameters that can be defined by the user. In this project, the scoring parameter was set to balanced accuracy due to the imbalance between the short and long class. The best possible parameter settings found using grid search for each algorithm are presented in tables 2 and 3.

DT	RF	SVM	MLP
Criterion: Entropy	# of estimators: 173	Kernel: RBF	Activation function: ReLu
Max depth: 10	Max depth: 20	Gamma: 0.001	Solver: SGD
Splitter: Best	Max features: Auto	C: 1	Alpha: 0.001
Max Features: log2	Criterion: Entropy		Learning rate: Adaptive
Min Samples Leaf: 4	Bootstrap: True		Layer Sizes: {90, 30}
Min Samples Split 10	Min Samples Split: 5		
	Min samples leaf: 1		

Table 2: The hyperparameter settings used for each base algorithm as a result of a grid search.

AdaBoost DT	AdaBoost RF	GradientBoost
# of estimators: 2000	# of estimators: 70	# of estimators: 200
Learning Rate: 1	Learning Rate: 1	Learning Rate: 0.1
		Criterion: MSE
		Min Samples Leaf: 2
		Min Samples Split: 5

Table 3: The hyperparameter settings used for each boosting algorithm as a result of a grid search.

3.8 Experiments

In this project, three experiments were defined in order to answer the research question and explore different approaches to the task of predicting LoS. The three experiments were conducted both on the ER Stage and the IA Stage.
The first experiment investigated the possibility of predicting LoS on the given dataset using the selected algorithms. The dataset was used in its full format and no additional pre-processing other than the one described above was applied. Each algorithm was trained and then tested on the test set. The results from this experiment are presented in table 4 and table 5 for the two stages.

In the second experiment, an exploration of different feature subspaces was conducted. By applying feature selection and incrementing the number of included features, the optimal number of features yielding the best performance could be found. For each algorithm, the hyper-parameters found from the grid-search were used as baseline model. The method for selecting the most relevant features was set to mutual information, which measures the mutual dependency between two variables.
For the algorithms chosen that supported feature weight or ranking in Scikit-Learn, Recursive Feature Elimination (RFE) was done to find the most optimal feature subspace and the most important features. RFE recursively removes the features with lowest feature importance in order to find the most optimal feature subspace. The results from the MI and RFE are presented in section 4.2.
Finally, the aim of the last experiment was to analyze how the subsampling ratio between the two classes affected the precision and recall for the long-staying class. In the experiments above, the ratio between the classes was one to one in all training cases. In this case, the subsampling of the short-staying class was incremented in the range 1400 contacts to 6000 contacts while the long-staying class was fixed. For each increment, the algorithm was trained on the training set and evaluated on the testing set.

4 Results

4.1 Experiment 1 - Base Dataset Performance

In the first experiment setting, each algorithm was trained at the two stages and evaluated on the testing set for each case. Evaluated metrics were balanced accuracy, precision for each class as well as training and prediction times.
At the ER stage, balanced accuracy ranged from 0.61 to 0.72 where Decision Trees had the lowest performance and SVM the highest. Precision for the long class ranged from 0.33 to 0.44 and for the short class from 0.84 to 0.89. Overall, SVM had the highest accuracy and precision in all categories. In terms of training time, AdaBoosted DT took the longest time at 25.7 seconds due to the large number of estimators. The fastest algorithm was Decision Tree. SVM had the longest prediction time.
At the IA stage, the predictive performance increased slightly compared to the ER stage. The balanced accuracy ranged from 0.65 to 0.75 with Decision Trees once again as the worst algorithm. Precision for the long class increased to a range of 0.4 to 0.48. The best algorithm for this stage was Gradient Boosted Trees which reached a balanced accuracy of 0.75, a long precision of 0.48 and a short precision of 0.91. Training time increased for SVM, Adaboosted DT and RF as well as Gradient Boosted Trees due to the higher number of features included.

4.1.1 ER Stage

Algorithm	Bal. Accuracy	Precision Long	Precision Short	Training Time (s)	Prediction Time (s)
Decision Tree	0.61	0.33	0.84	0.02	0.02
Random Forest	0.70	0.43	0.88	3.27	0.13
SVM	**0.72**	**0.44**	**0.89**	7.61	3.67
MLP	0.7	0.42	0.88	11.5	0.02
AdaBoosted DT	0.70	0.43	0.88	25.7	2.75
AdaBoosted RF	0.71	0.43	0.89	10.8	0.33
GradientBoosted Trees	0.71	0.44	0.89	14.9	0.02

Table 4: Accuracy, precision, train and test times in the first experiment setting using ER Stage data. Evaluation was made on the testing set.

4.1.2 In-Admission Stage

Algorithm	Bal. Accuracy	Precision Long	Precision Short	Training Time (s)	Prediction Time (s)
Decision Tree	0.65	0.4	0.85	0.02	0
Random Forest	0.74	0.46	0.91	2.69	0.09
SVM	0.74	0.46	0.91	8.09	3.78
MLP	0.71	0.43	0.89	7.33	0.02
AdaBoosted DT	0.73	0.46	0.89	28.33	2.83
AdaBoosted RF	0.74	0.46	0.91	12.38	0.3
GradientBoosted Trees	**0.75**	**0.48**	**0.91**	19.3	0.02

Table 5: Accuracy, precision, train and test times in the IA Stage setting. Evaluation was made on the testing set.

4.2 Experiment 2 - Feature Subspace Selection

Experiment 2 investigated how the performance was affected by the number of features included in the feature set. Two approaches were evaluated for both stages.
The first method was based on selecting features with Mutual Information and incrementing the number of included features. The results were that a drastic increase in performance could be seen for the first 75 features. After 75 features, no performance gain could be seen. For the boosting and ensembles methods, the performance remained at the same level for feature sets including more than 75 features. However, for SVM and MLP, a slight decrease in performance could be seen as the feature set continued to grow. MLP peaked at around 120 features and SVM at 130 features. Decision Tree had no clear pattern with increased number of features.
In the second approach, Recursive Feature Elimination was used to strip away unnecessary features until the optimal number of features was found. This method was only available for certain algorithms in scikit-learn. Similar evolution of the performance compared to the mutual information feature selection could be seen here as well. Gradient Boosted Trees and Random Forest quickly reached a maximum accuracy of 0.71 for the ER stage and 0.72 for the IA stage which then remained quite stable for increasingly larger feature sets.

4.2.1 Feature Selection at ER Stage

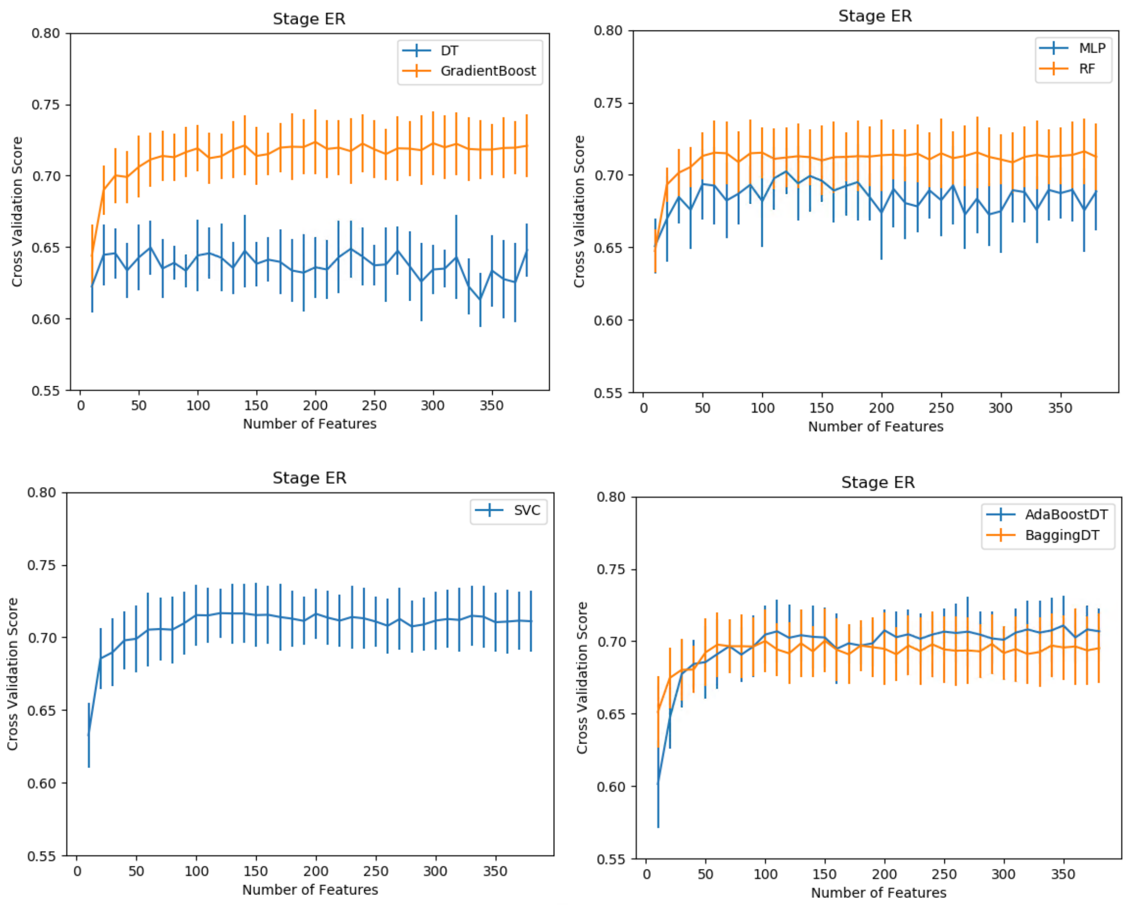


Figure 5: Mean cross validation score plotted over varying number of features selected using Mutual Information at the ER Stage. At each iteration, the K best features are selected and evaluated on.

4.2.2 Feature Selection at IA Stage

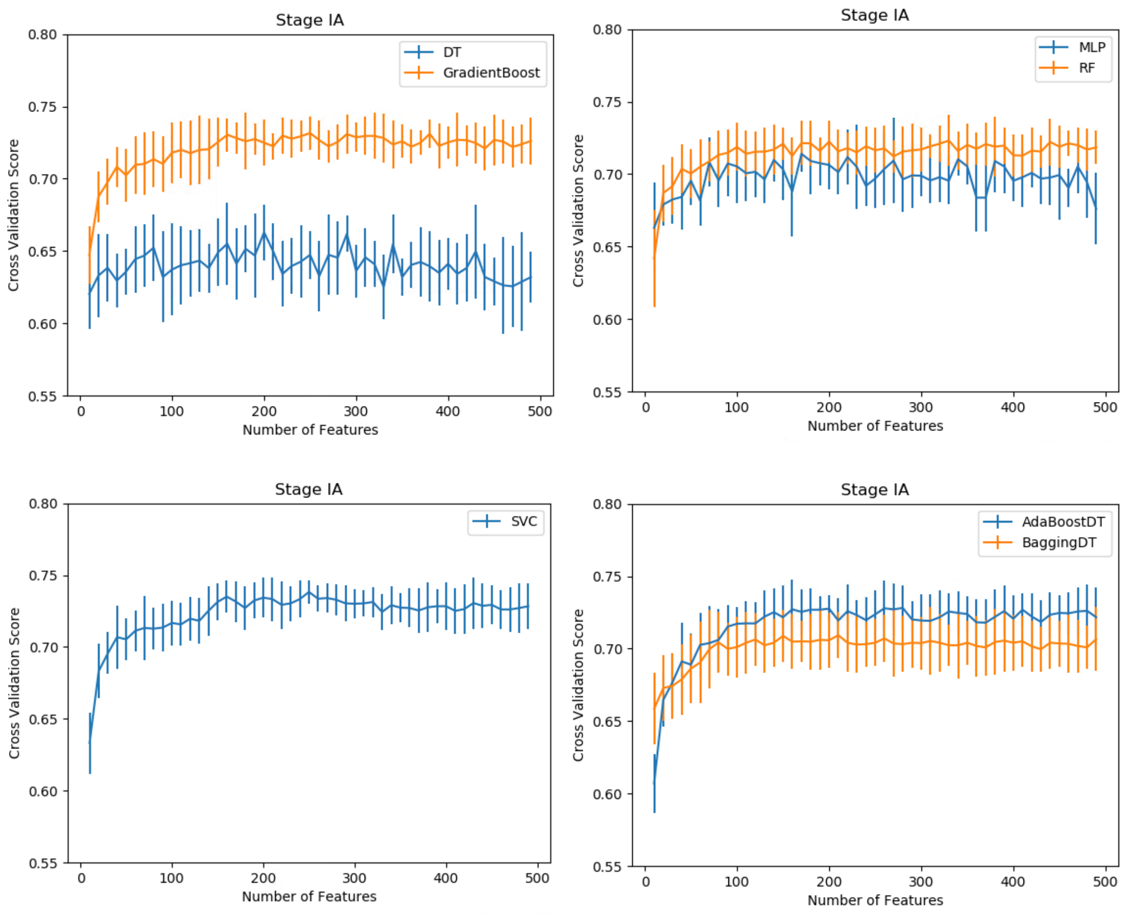


Figure 6: Mean cross validation score plotted over varying number of features selected using Mutual Information at the IA Stage. At each iteration, the K best features are selected and evaluated on.

4.2.3 Top Mutual Information Features

Top 20 MI Features			
ER Chemistry Test 0	ER Chemistry Test 150	ER Chemistry Test 197	ER Chemistry Test 25
ER Chemistry Test 83	IA Chemistry Test 5	IA Last Pulse Measure	IA Mean Oxygen Saturation
Mean Previous LoS	Median Previous LoS	Most Recent LoS	Number of Diagnosis
Number of Prescriptions	Number of Previous Admissions	Shortest Previous LoS	Total Previous LoS
ER Chemistry Test 4	Longest Previous LoS	Number of Earlier Procedures	Admission Reason 672938

Table 6: The top 20 features extracted using mutual information, not in ordered fashion. Note that the feature naming is in accordance with the anonymization method used in this project and hence the specific names of tests are not available.

4.2.4 Recursive Feature Elimination

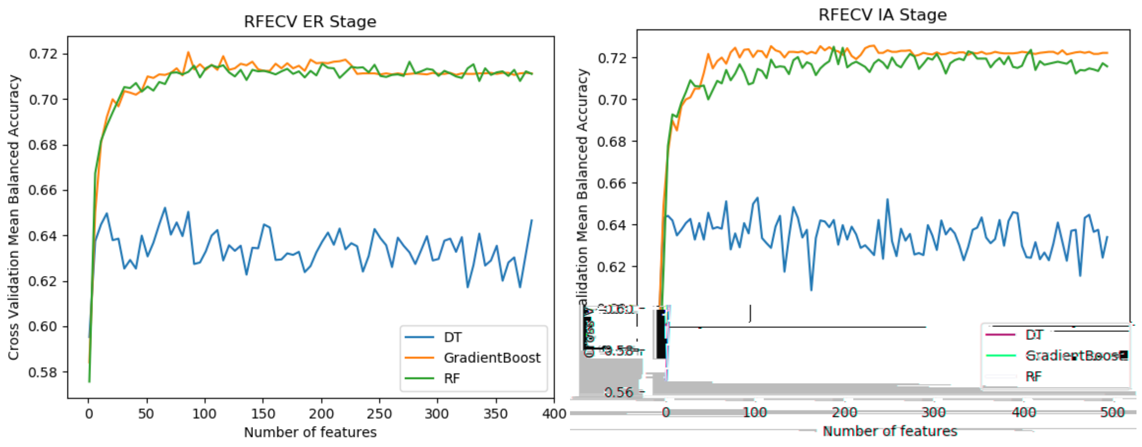


Figure 7: Mean cross validation score plotted over varying numbers of features when running the RFE. Note that despite AdaBoost supporting RFE, there were memory issues with these algorithms resulting in them missing from this analysis.

4.3 Experiment 3 - Subsampling Ratio Tuning

The last experiment investigated how the precision and recall changed with the number of included samples from the short class in the dataset. For each algorithm at both stages it could clearly be seen that a trade off between recall and precision for the long class was possible to achieve with increased number of short-staying contacts. As the number of short samples increased, the precision in the long class increased slightly while the recall dropped more drastically.
The confusion matrices in fig. 10 and fig. 11 shows three different points in the subsampling process at each stage.

4.3.1 ER Stage

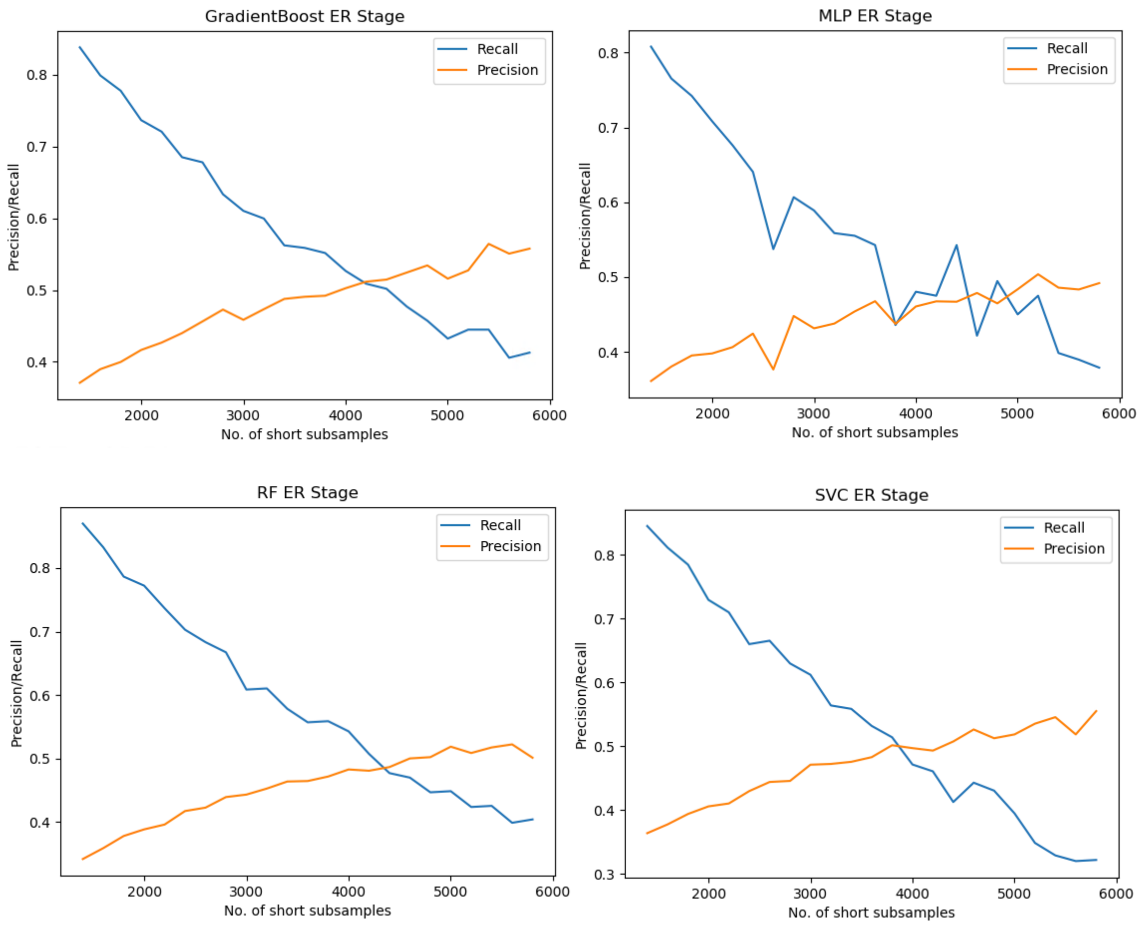


Figure 8: Precision and recall for the long-staying class evaluated on the testing set plotted over number of short-staying contacts included in the training set.

4.3.2 IA Stage

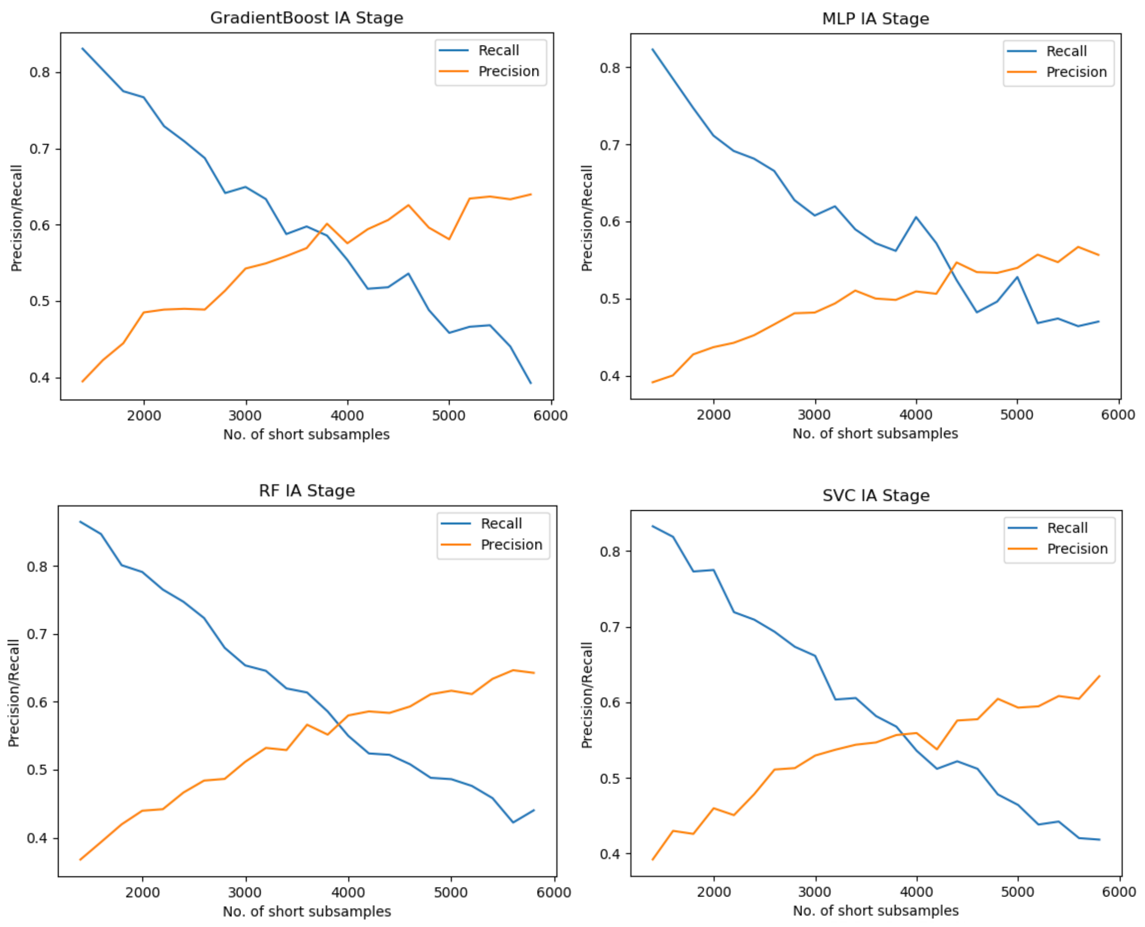


Figure 9: Precision and recall for the long-staying class evaluated on the testing set plotted over number of short-staying contacts included in the training set.

4.3.3 Confusion Matrices - ER Stage

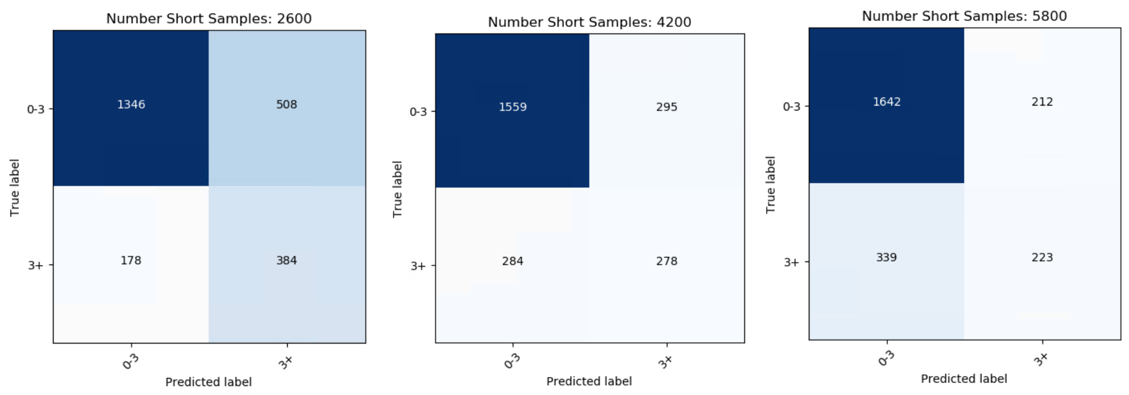


Figure 10: Confusion matrices from the test set evaluation with larger portion of short-staying contacts when subsampling. The algorithm was Random Forest.

4.3.4 Confusion Matrices - IA Stage

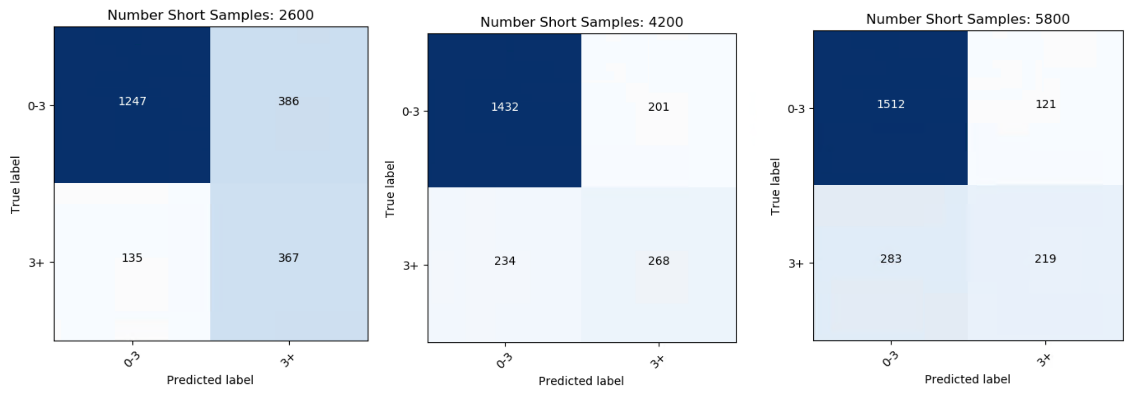


Figure 11: Confusion matrices from the test set evaluation with larger portion of short-staying contacts when subsampling. The algorithm was Random Forest.

5 Discussion

5.1 Model Comparison

The main aim of this project was to investigate if it is feasible to predict LoS using machine leaning and secondly, to investigate and compare different ML models for the task. From the results presented in section 4.1, it can be concluded that it does in fact seem possible to predict patient length of stay at time of admission to the ward from the ER.
It can be seen in table 5 that by delaying the prediction somewhat into the admission time, a better prediction can be made. This result is in line with the hypothesis that complementing the data from the ER with the data collected at the ward would increase performance. Important features such as suspected diagnosis and planned surgery are added at this stage and this type of specific information can be valuable, especially when combined with all the lab parameter values. All models seem to perform roughly equally well except stand-alone decision tree. For the ER stage, the balanced accuracy ranges from 0.7 to 0.72 excluding DT and for the IA stage this range is 0.74 to 0.75. These results were in line with the ones obtained in previous LoS studies.
The fact that the patient group used in this dataset was quite broad shows that there is a potential for these ML algorithms to be used in more generalized settings in hospitals. Earlier research have usually very specific limitations on patient type, such as diabetes or brain-surgery patients, while this project focused on a certain hospital clinic.
One of the limiting factors was the low precision for the long-staying class (i.e. patient contacts staying longer than 3 days). At the ER stage, this value ranged from 0.33 to 0.44 and for the IA stage it ranged from 0.4 to 0.48. Such a low precision would make it troublesome to use the prediction in a real system. Ideally, a precision over 50 % would be sufficient and judging from the improved results at the IA stage, a valid method would either be to delay the prediction further or add more data.
It is difficult to draw any conclusions about why the long-staying class has such a low precision, but manual investigation using plots of combinations of numerical features (vital parameters, chemistry values etc.) showed that it seemed hard to find a good representation in which the two classes were easily separable. For certain categorical variables such as specific diagnosis or arrival reasons, it was easier to see a clear separability in LoS. The problem for this type of data was that there was usually a lack of patient contacts with that specific code, making it hard to draw any conclusion from it about the whole population. As a result, it seemed that even though the algorithm could be able to predict long-staying patients well (i.e. most true long-staying patients were actually predicted as such), there would always be an overlap with certain short-staying patients which resulted in poor precision.

5.2 Feature Selection, Important Features And Subsampling Ratio

From the two methods of feature subspace exploration, it could be concluded that reducing the feature subspace did not necessarily increase the performance. However, it could be seen that a smaller subspace than the one provided in experiment 1 could be used and still reach an equally good performance.
These findings indicated that there was a substantial part of the dataset that did not provide new, meaningful information to the algorithm. From looking at the graphs from figure 6, it could be seen that using around 50 features almost resulted in the same performance as 300+ features. Upon further inspection of what these top 50 features consisted of when filtering with MI, it was seen that essentially all these features were numerical ones with a few categorical features such as prescriptions, marital status and age interval. This is partly presented in table 6.
This result was interesting because it meant that a large part of the dataset that consisted of very specific information such as earlier diagnosis and procedures was excluded or redundant. One

possible reason for this could be that each specific code for diagnosis and procedure had a very small support in the dataset, i.e. only a few patient contacts had the code, resulting in that no conclusions could be drawn regarding LoS if a patient had the code or not.
One way to bring up the information content in diagnosis and procedure codes would be to investigate different coding systems. In the case of medications, it was not feasible to include each specific code available as explained in section 3.3.3 but instead, a higher grouping with the help of the ATC encoding was used. A similar approach could be made with the diagnosis codes using for example the ICD (International Statistical Classification of Diseases and Related Health Problems) encoding or the KVÅ (Klassifikation av vårdåtgärder) encoding in order to leverage the information content in these features better.
From the third experiment one can see that by varying the subsampling in the short-staying class it is possible to tune the trade-off between precision and recall for the long-staying class. Naturally, including a larger part of the dataset should mean that the algorithm performs better.
Considering that the precision for the long-staying class was so low from experiment 1, this method provides a way to actually increase the precision and therefore also the usability of the algorithm. This type of tuning could be offered to the physician when selecting an algorithm for implementation in order to let him or her adjust the trade-off and select a model that suits their needs best.

5.3 Usability in Healthcare

The usability of an ML algorithm is first of all usually based on its performance. The limiting factor for the models investigated in this project in terms of performance was, as discussed above, the low precision. Despite this, there is a large potential for a model to be used on the data that was available in this project given that the restrictions on the dataset were very strict.
One of the main limitations in this project was the heavy data anonymization applied to the dataset. In a real hospital setting, running the algorithm on-premise, there would be more freedom experimenting and manipulating the data meaning that the performance could potentially be improved further. A few ways that could be explored in an un-anonymized setting would be more extensive use of dates and time stamps (in this thesis, only the admission weekday was included) as well as more specific analysis of the different codes. One could for example include codes that the physicians might think have a large impact on LoS, instead of just selecting the ones with the largest support as was done in this project.
From the aspects considered in table 7, it could be argued that the models used in this project are suitable for integrating into the healthcare workflow. In terms of timings, both training time and prediction time are fairly quick. Most models used here, with the exception of AdaBoost with a large number of estimators, could be trained quickly with limited resources. This means that running the algorithm on-premise on for example a local server could be a valid option, allowing for flexible implementation and cost-efficient deployment.
A key factor would be the interpretability of the model. When running and comparing ML models in an experimental setting, interpretability is not as crucial. However, when aiming for a solution in the healthcare setting, the algorithm would be exposed to non-technical staff in the form of physicians and nurses. In order to motivate why the ML model can perform as it does, the algorithm should be easy to understand and transparent.
The models investigated in this project are not the simplest ones available, especially not in terms of how the algorithms work and in the relationships they try to model. They do however support useful tools such as feature importances and prediction probabilities, meaning that a solid framework of presenting why a prediction was made as it was could be developed. Decision Tree, which is a very intuitive model and could be considered one of the most interpretable models used in this project, performed fairly poorly compared to the rest of the algorithms. Judging from table

4, the most suitable algorithm for prediction at the ER stage would be Random Forest. Even though SVM performs slightly better, the prediction time is longer. This could be problematic if many real-time predictions need to be made close in time. Another issue is that SVM with the RBF kernel does not provide any sort of explainability, reducing the interpretability of this model severely. RF provides good performance while quick to train and predict. It is also more intuitive to understand than SVM and also provides the necessary features from an interpretability perspective, i.e. feature importances.

6 Future Work And Limitations

There was, as mentioned in the method, large amounts of anonymization applied to the dataset which also was one of the biggest limitations in this project. Without knowing the names of chemistry tests and diagnoses, it was hard to do any sort of clinical analysis and to use suspected correlations to make the feature engineering easier.
Furthermore, the dataset available in this project contained a lot of information that was not included in the training and testing data due to the amounts of feature engineering it would have required. As the whole pipeline from raw data to polished flat-table format had to be implemented, not all possible information could be extracted due to time limitations.
This is one of the most interesting points to explore in similar projects in the future. The dataset contained copious amounts of time-series data of different lab tests and it would be very interesting to develop more features connected to trends present in these types of data. Only static features were used in this project, but if features could be extracted that reflect how for example a vital parameter has varied over time, this could prove very valuable to the ML models as trends are very important when clinicians evaluate patients. It would therefore be interesting to more extensively explore the feature engineering aspects of this dataset, not only for the time-series data.
The number of models evaluated in this project was also limited and selected on the basis of earlier research. There are other possible models to explore and one interesting method could be to investigate how deep learning performs on the available data. Deep learning networks can also be used in other ways than for prediction. For example, it could be interesting to investigate how so called partial variational autoencoders (partial VAE) could be used to handle missing data. This type of autoencoder transforms incomplete data representations into another one that an ML model still can use, efficiently recovering missing information [5].
Finally, an interesting method of evaluation used in [6] could be investigated. Along with evaluating against the actual outcome, this paper included physicians predictions and evaluated how the ML algorithm performed against these man-made predictions. As hospitals use these manual predictions already, it would be of interest to see if the algorithm performs better. If that is the case, then an ML tool could still prove more useful than the existing method and could be worth implementing despite the precision issues found in this project.

7 Conclusions

This project has investigated the possibility of predicting length of stay for patients admitted from the emergency room to a hospital ward at a Swedish hospital. The prediction problem was formulated as a binary classification with classes shorter or longer LoS than 3 days. The results showed that it was possible to achieve a balanced accuracy of of 72 % on the testing set when predicting at the time of admission from the ER. One problem was the low precision for the longer class. However, it was shown that this could be tuned by varying the amount of subsampling in the short-staying class that was heavily over-represented in order to achieve a better precision. The final, recommended algorithm was Random Forest. This model provided reasonable accuracy, quick training and testing speed and high interpretability which would allow for a light-weight implementation using limited hospital resources.

A State of The Art Analysis

A.1 Length of Stay

Length of stay (LoS) is a term used to describe the duration of a patients hospitalization. It is commonly defined as the difference between the admission date and the discharge date of the patient to and from the hospital. However, some definitions do not count discharge to another hospital or ward as the end of the LoS. LoS is used as a qualitative measure for several purposes. It can be important from both a clinical and administrative perspective. For clinical use, LoS can be interesting to measure in order to evaluate if patients receive the right care and if longer hospital stays can affect their rehabilitation. Such an approach has been studied in [7] where it was shown that a decrease in LoS for geriatric patients did not affect the activity and mobility of the elderly patient after discharge. Another study showed that by correlating certain features, such as patient treatment or medication received during the stay, with LoS it was possible to identify the most effective medicine for pneumonia treatment as well as other factors highly correlated with LoS [8]. Those factors can then be included in the optimization and re-evaluation of hospital routines and patient flows.
More typically, LoS is studied from an administrative perspective and often in correlation with hospital costs. Hospitals have limited resources in terms of both staff and beds which means that patients with longer LoS can have serious implications for the hospital in different ways. It can not utilize its full capacity, extra resources may have to be focused on the long-staying patient and other procedures for other patients are hindered due to the lack of beds. Hence, LoS can be a valuable tool for administrative reasons in both bed planning and cost savings because it allows the staff to plan ahead and where to coordinate extra resources.
Several studies have been done on LoS and its effects on patient costs. Such a study is presented in [9] in which a multilevel modeling approach is made to examine patient costs and how they relate to both patient and hospital features. The results show that a reduced LoS considerably reduced the hospital costs.
Another study showed that an increased LoS was one of the main drivers of increased costs for patients that had undergone craniotomy for tumor resection [10]. This report also discusses that the emphasis of LoS reduction should not be focused on general patients, but rather to identify the cases that result in a long LoS but where the patient comorbidities are not justified for such an outcome. It is important to keep this aspect in mind when studying LoS. The care of the patient must not suffer when actions are implemented to reduce the overall LoS.

A.2 Predicting LoS

Because of its popularity as an evaluation metric for hospitals there has been an increased interest in predicting the LoS for patients being admitted to a ward or clinic. Clearly, hospitals are under increasing pressure as the population increases while the available resources such as staff, money and space are reduced. If a reliable prediction of LoS could be made at admission time, it could serve as a valuable tool for nurses, doctors and administrative personnel at the ward. It could for example allow doctors to more efficiently schedule surgeries as they could have an estimate on when certain patients will be discharged or how long they potentially will occupy a bed. Furthermore, a prediction could serve as a clinical indicator that a patient should receive more thorough care or special treatment if his or her earlier medical history indicates a long LoS.

A.3 Statistical Modeling of LoS

The first approach to model and predict LoS is to use statistical modelling and find correlating factors that significantly contribute to an increased LoS. These type of models can give hints about

what potential factors might need to be monitored in order to identify long-staying patients, but they might also produce their own predictions.
In [11], a statistical analysis is presented in which several factors were studied using t-test, analysis of variance (ANOVA) and other statistical tests in order to find whether or not they had a significant correlation with longer LoS. Some of the factors identified were if a patient was admitted from an outpatient setting such as nursing home or if they had low systolic blood pressure at admission time. One interesting aspect of this study is that the authors present suggested actions for the three main contributing factors that can be applied proactively. By presenting such proactive interventions to the hospital, this type of statistical study can be of high value when re-structuring hospital routines.
More complex statistical modelling of LoS can be done using for example Markov chains. Such an approach is presented in [12] which tries to model the average LoS in an intensive care unit (ICU) using Markov chains. The results presented are quite vague and the estimations from the Markov chain are not particularly good. The strong points are that the model can give estimates about to which specific part of the ICU the patient will transfer to and an estimate of the discharge time. The problem using such complex modelling methods are that physicians might not understand the processes that makes the model valid and hence doubts the estimates given. This is a major factor to consider when trying to take the step from research environment to an actual healthcare setting.

A.4 Machine Learning Approaches to LoS

The other approach and the main topic of this thesis is to use machine learning (ML) to predict patient LoS. Machine learning methods have received an increased interest since improved processing power, access to data and readily available machine learning libraries such as scikit-learn has made these type of methods easily accessible. This development has of course been seen in the area of LoS prediction as well and has shown promising results.
ML algorithms can help identify factors much like the statistical methods presented above but also predict the LoS for a given patient by using data available at the admission time. This is done by training such a model on previous seen cases and for each iteration update the algorithms parameters in order to minimize a loss function or some other optimization criteria. This process is usually one of two types. *Supervised learning* in which the correct answers (for example the actual LoS in this case) is known or *unsupervised learning* in which no correct answer is known and the task is to find some correlation or pattern to group the data.

A.5 Problem Formulation

This section focuses on how the problem of predicting LoS can be formulated along with how earlier studies have approached the problem. Working with healthcare data poses many challenges. How should missing data be treated, how do one handle inconsistency within the data and should certain medical features be treated differently? By clearly structuring a data processing pipeline and machine learning workflow, one can avoid these problems.

A.5.1 Classification And Regression

Prediction of LoS is mostly tackled as a task for supervised learning and can be approached as two different types of problems. The first approach can be to try and classify patients into categories such as LoS shorter or longer than a certain amount of days. This is called a *classification* task because the aim of the algorithm is to predict a new, unseen sample into a specific class. The other approach is to predict a continuous value which could for example be to predict an exact LoS in terms of hours or days. This type of prediction is called *regression*.

What approach is most suitable depends on the problem setting. In a hospital it might be seen as to precise to regard the LoS as a regression problem and any information that can be given by a classification algorithm suffices for efficient resource management. In other cases it can be useful to know if a patient will potentially be discharged in the morning or afternoon. Hence, most research evaluate both types of approaches as there might be differences in performance, training time and needed computation power which are also important to consider if the algorithm is to be used in a healthcare setting. A more exhaustive table of important evaluation criteria that have been presented in earlier research along with other common criteria is presented in table 7.
One aspect if classification is the chosen method is the splitting of data into classes. Because LoS is typically distributed with a large majority of patients having a shorter LoS then followed by a long, smaller tail, picking suitable classes can be difficult. In [13], the authors split the data into two classes: shorter-than-three-days and longer-than-three-days. They motivate this split just because the LoS distribution changes significantly after two days and that this would indicate that the patients with longer LoS than that could require more extensive planning.

A.5.2 Data And Feature Pre-processing

A large part of machine learning consists of processing available data and extracting relevant features that help the algorithm perform better. This part is especially important when handling medical and patient data. If free-text data is included in the dataset, potential private information regarding the patient might be disclosed resulting in serious loss of privacy for the individual and violation of laws such as GDPR.
Most studies focus heavily on one patient type or category. For example, in [14] the authors focus on cardiology patients, in [13] the focus is on diabetic patients and in [15] the patients have all undergone brain tumor surgery. By limiting the patient type, the data pre-processing becomes easier as the data collection and structure is more standardized for that specific ward. Also, generalizing an ML model to several wards or even hospitals would require enormous amounts of data. These databases are usually highly protected for privacy reasons or available in the US only.
Imputation of medical data is also something that has to be done carefully. Any eventual imputation of missing data has to be very well motivated from a clinical viewpoint. An interesting example of imputation consideration is presented in [16]. The approach for imputing data was that any feature with more than 50 % missing data was removed and considered not effective for analysis. If a feature had less than 12 % missing data, the value was replaced either with mean or mode depending on feature type. For features with more than 10 % missing data, an interesting approach of fitting a decision tree to that feature was implemented in order to fill in the missing value with predictions from the tree.

A.5.3 Evaluation Methods

Evaluation of algorithm performance is a very important aspect. In the case of classification, most common metrics are accuracy, precision and recall. Accuracy is the most straight-forward measurement but can be misleading if classes are not equally large. In those cases, it is better to consider recall/precision for each class instead. They are defined as.

$$Accuracy = \frac{\#\, of\, correctly\, classified}{\#\, of\, total\, samples} \tag{1}$$

$$Precision = \frac{tp}{tp + fp} \tag{2}$$

$$Recall = \frac{tp}{tp + fn} \tag{3}$$

Evaluation Criteria	**Stakeholder**	**Comment**
Accuracy/Error	Physicians, Nurses, Data Scientist	One of the main evaluation metrics used for evaluating the performance of an ML algorithm along with similar measures. Must be one of the main consideration when applying the algorithm in the hospital.
Sensitivity & Specificity	Data Scientist, Physicians, Nurses	It is important to evaluate how the algorithm handles false positives/negatives. What impact does a misclassification have on the patient and the hospital organization?
Interpretability	Physicians, Nurses	How easy is the algorithm to understand? Is it easy to understand the reasoning behind a prediction? Especially important in healthcare where important clinical decisions are made and always needs to be well motivated.
Training time & Resources	Data Scientist, IT-administrator	An algorithm must be feasible to train in an healthcare setting. Deep learning solutions requiring heavy GPU clusters and long training time might not be practical to employ.
Prediction time & Resources	Data Scientist, IT-administrator	Prediction with the algorithm must be fast enough to provide almost real-time updates. A ML algorithm must not interrupt the workflow at the clinic or ward.
Dataset	Data Scientist	Amount of training data needed for the algorithm to perform well. Algorithms requiring too much data might not be possible to use.
Robustness	Data Scientist	ML algorithms are usually trained for ward-specific solutions. Different patient flows, countries, patient types and other factors make generalization of an ML algorithm hard.

Table 7: Important evaluation criteria and their specific meaning for the healthcare setting as well as what potential actors in the process they can be of use for.

If regression is the method of choice, the distance between the predicted and true value is the measure of error. Two common ways are Mean Absolute Error (MAE) and Mean Squared Error (MSE), defined below. MSE penalizes larger errors more and considerations have to be made before picking a suitable regression error metric.

$$MAE = \frac{1}{n}\sum_{i=1}^{n} |(Y_{true,i} - Y_{pred,i})| \tag{4}$$

$$MSE = \frac{1}{n}\sum_{i=1}^{n} (Y_{true,i} - Y_{pred,i})^2 \tag{5}$$

A.6 Machine Learning Algorithms

Research on LoS prediction using machine learning have been using a multitude of different approaches and different algorithms. The aim of this section is to provide an overview of the most commonly used algorithms which appear in most papers and have been proved to perform well.

A.6.1 Linear & Logistic Regression

Linear regression is a method that can be used to describe the linear relationship between one dependent variable (the outcome variable) and one or more independent variables (features). The relationship should be linear for the model to work and gives a continuous output [17]. A linear regression model can be, given features x_1, x_2 fitted with coefficients b_0, b_1, b_2, represented as

$$y = b_0 + b_1 x_1 + b_2 x_2 \tag{6}$$

Logistic regression is similar to linear regression in many aspects. The main difference is that in logistic regression the aim is to estimate a probability of a sample belonging to one of two classes. Logistic regression produces a logistic curve based on a set of estimated coefficients. If there are two features x_1, x_2, the logistic function can be described by the fitted coefficients b_0, b_1, b_2 using

$$p = \frac{1}{1 + exp(b_0 + b_1 x_1 + b_2 x_2)} \tag{7}$$

In which p is the probability that the sample belongs to the class or not [18]. Logistic regression is used for binary classification problems.
Both linear and logistic regression are mostly used as baseline models which is the case in [6]. In this study, the aim was to predict if a patient will be discharged the same day and logistic regression was used as a baseline comparison. The performance was a sensitivity of 65.9 % and a specificity of 52.8 %. Another study used linear regression to develop a predictive model for the prediction of LoS in burn victims [19]. This model managed to reach a R^2-score of 0.2 which the authors deemed good.
Besides its important role as a baseline model, linear and logistic regression can prove to be valuable in other aspects as well. The estimated coefficients in linear regression serve as an easy-to-interpret representation as the final linear formula can be understood by most professions. The drawback is that most relationships are complex and when applied to big datasets with many features the linear regression might not catch important non-linear patterns.

A.6.2 Decision Trees

One of the most traditional approaches to LoS prediction is to use a decision tree. A decision tree can be seen as a tree-like structure which at each node splits the dataset by looking at a certain feature or features. For example, a split in can be made by looking if the age is above or below some threshold. The aim of the tree is to traverse the nodes, looking at a feature and making splits and ultimately end up at a leaf-node in which a classification or prediction is made.
When building the decision tree, these splits are not simply done at random. When attempting a split, the decision tree tries to optimize the split in regards to some optimization criteria. Usually, this criteria is set as an impurity measurement of the feature. If a feature can take on one certain value with probability 1, then that feature has no impurity. On the other hand, if the possible values for a feature each have the same probability, the impurity is maximized. A measure of impurity that is commonly used is *entropy* which is defined as

$$Entropy = -\sum_i p_i log_2 p_i \tag{8}$$

In which p_i is the probability of class i and we sum over all the possible classes. This measurement is used in what is called the *information gain* which is defined as

$$Gain(i, X) = Entropy(i) - Entropy(i, X) \tag{9}$$

In which i is the class and X is the feature to make the split for. Information gain is one of the most common splitting criteria used, but other types of criteria are available as well [20], [21].
Not many papers apply the decision tree as a stand-alone model. More commonly, it is used in an ensemble method which is explained later on. One study that does apply a single decision tree is presented in [16]. In this study, a dataset of 3266 patients with 36 features each was available and the aim was to classify each patient into three categories of LoS. These categories were LoS between 0-5 days, 6-9 days and longer than 10 days. The decision tree resulted in a classification accuracy of 83.5 %. Compared to the other methods evaluated in this paper, SVM and ensemble methods which both achieved over 95 % accuracy, this performance was not very good but the authors still motivates its use by the easy interpretability of the graphical representation of the tree.
The grouping of LoS into three different classes is a useful approach. It provides the healthcare staff with the right amount of information while reducing the ML problem to a somewhat simpler problem. Usually, this sort of approach is also good due to the skewness of LoS data. LoS data is typically skewed towards the shorter side, something that proves problematic for many ML algorithms, and splitting the dataset into roughly equal classes can help leverage this problem [22] as mentioned in section A.5.1.

A.6.3 Support Vector Machine

The support vector machine (SVM) is an algorithm that regards each data sample as a point in a high dimensional space in which each feature is regarded as an axis or dimension in this space. The aim of the SVM is to fit a hyperplane that best separates all the input points in the space into two different classes. This task is done by solving an optimization problem.
Commonly, SVM uses a transformation called the kernel trick. The kernel trick is used to project the input points into a higher dimensional space in which they might be easier to separate than in the low-dimensional space the data is originally represented in. The advantage of the kernel trick is that it is not necessary to compute the transformation to the higher dimensional space for each point, but only to compute the inner product between the points in that space which saves a considerable amount of computations. This trick allows the SVM to model very complex and non-linear relationships [23], [24].
In LoS prediction, SVM is used both for classification and regression. In a study of 29 different ML algorithms that were to be evaluated and combined into an ensemble predictor it was found that a SVM was in the top three best performing algorithms [15]. A variant of the SVM was also the top performing algorithm in [13] with an accuracy of 0.68.
With its ability to capture non-linear, complex patterns the SVM makes for a strong algorithm. The basis of the SVM is intuitive to understand to a certain degree but in terms of interpretability it is certainly more limited than the decision tree. On the other hand, choosing the right kernel requires expertise and the parameter tuning might prove difficult.

A.6.4 Artificial Neural Networks

Artifical neural networks (ANN) have received increased interest in many fields over the last years. These types of networks tries to mimic the neurons in the brain to a certain degree. By interconnecting layers of neurons where each neuron has a weight and a bias as well as an activation function, the neural network can estimate non-linear decision boundaries very well.
Neural networks have excelled in several tasks. One specially interesting variant of neural network is what is called a deep neural network. The deep neural networks is essentially a regular network that consists of considerably many more layers of neurons than the regular ones. Deep learning

have been successfully used in segmenting urban driving environments, various semantic segmentation tasks and medical image segmentation [25], [26].
Artificial neural networks have been used in predicting length of stay with mixed success. In [14] prediction of LoS for three different types of diagnoses using ANN was investigated. A model was trained for each separate diagnosis and also for two cases, one using all clinical data available from the hospitalization called predischarge stage and one using only prior data called preadmission stage. 21 features were available for 2377 patients. The results were than no significant difference in performance could be seen between the preadmission and predischarge stages. For one of the diagnoses, a mean absolute error (MAE) of 1.03$\tilde{1}$.07 was achieved, while for the other two the MAE was about 3.87. For the latter two diagnoses, the linear regression outperformed the ANN in both stages.
The authors reason that more features, such as lab values and vital signs, would need to be included to improve performance but also that a categorization of patients into risk groups might have been more meaningful.
Perhaps one of the more ambitious studies is conducted in [27]. This study applied deep learning onto a dataset of 216 000 hospitalizations in order to predict LoS, risk of mortality, risk of readmission and also inference of discharge diagnosis. The evaluation metric was area under the receiver operating characteristic curve (AUROC) and the model performed 0.86 for one hospital and 0.85 for another. This outperforms most other models previously presented. This was possible due to extensive utilization of the electronic health record (EHR) and all the features and free-text information available in it. While a main concern for neural networks is the lack of explainability, the authors present a way to show relevant free-text sections that contribute to the prediction which allows physicians to judge if it is credible or not.
The drawback of this ambitious study is the need for immense computer power. Training deep learning models require both large amounts of training data but also massive GPU resources which makes it both difficult and expensive to develop these models. While the performance of the model is outstanding, these factors limits the possibility for hospitals to develop and deploy such algorithms themselves. Another critical point is the use of free-text EHR information, something strict privacy regulations such as GDPR limit and can prove hard to get access to.

A.6.5 Ensemble methods

All the algorithms presented so far perform as a single model. In order to improve model performance it is common to combine many algorithms into what is called an ensemble. There are different ways and examples of doing this and usually such a ensemble algorithm is called a meta-algorithm. The first method is to use bagging. In bagging, subsets of training data are generated by randomly sampling the dataset and used to train a single model. The trained models are then combined and the output is given as an average of all predictors [28]. This reduces the variance of the predictor and gives a more robust model.
An extension of bagging is Random Forest (RF). RF uses decision trees trained not only on random subsets of data, but also random subsets of features in that data. RF is a popular algorithm for LoS prediction and has been used in [6] for regression and end-of-day discharge. It was shown that RF could perform with a sensitivity of 60% and specificity of 66%. In that study, an interesting comparison with physicians predictions was made as well. In terms of predicting end-of-day discharge at 2 p.m. the same day, the physician predicted 485 cases correct while the model managed to predict 1781 cases correct.
RF is also the preferred model in [29] due to its interpretability and its capability to capture non-linearity. It was trained on 44 000 hospitalizations and performed on the test set a mean squared error of 2.26.
Another meta-algorithm is called boosting. In contrast to bagging, which is based on parallel learning of several different models, boosting fits models sequentially. It begins by fitting a model

to the original data and based on the misclassification of that model, weights are assigned to hard-to-classify samples which are then used in the training of the next model. At the end, an ensemble of weak learners is achieved that performs well using the weighted average based on the weak learner performance [30].

ML Algorithm	**Advantages**	**Drawbacks**	**Earlier results**
Linear Regression	Very easy to interpret. Baseline model.	Only fits a linear model. Not good with large feature space.	For predicting end-of-day discharge: Sensitivity 65.9% and specificity 52.8% [6]. Predicting LoS for thrombectomy: RMSE of 8.764 [31].
Decision Tree	Very interpretable and intuitive. Easy to train. Non-linear.	Can easily overfit.	Accuracy of 83.5% [16]. Predicting LoS for burn patients: MAE of 7.192 [32]. Prediction of LoS in colorectal cancer: accuracy of 72.32% [33].
Random Forest	Non-linear. Easy to interpret. Ensemble method reduces variance and overfit.	Several hyper-parameters to tune in order to perform well. Slightly slower to train than D-tree.	Predicting end-of-day discharge: Sensitivity 60% and specificity 66%. Predicting LoS for diabetic patients: Accuracy of 65% [13].
SVM	Good ability to capture non-linear, complex relationships.	Hard to find good hyper-parameters. Kernel selection requires expertise. Slow to train.	Prediction of LoS in colorectal cancer: Accuracy of 73% [33]. Predicting LoS for diabetic patients: Accuracy of 68% [13].
ANN	Good ability to capture non-linear, complex relationships.	Low interpretability. Lots of training data.	Prediction of LoS in colorectal cancer: Accuracy of 71.2% [33].
Deep Learning	Proved very good performance. Same advantages as ANNs	Requires computational power. Hard to interpret. Lots of training data.	Predicting LoS: AUROC of 0.81 [27].

Table 8: Summary of the most commonly used ML algorithms for the LoS prediction task, their advantages and disadvantages in broad terms as well as previous performance.

References

[1] Department Population and Welfare. *Rapport: Sveriges Framtida Befolkning 2018 - 2070.* Statistics Sweden (SCB), 2018.

[2] Hayit Greenspan, Bram Van Ginneken, and Ronald M Summers. "Guest editorial deep learning in medical imaging: Overview and future promise of an exciting new technique". In: *IEEE Transactions on Medical Imaging* 35.5 (2016), pp. 1153–1159.

[3] David W. Bates et al. "Big Data In Health Care: Using Analytics To Identify And Manage High-Risk And High-Cost Patients". In: *Health Affairs* 33.7 (2014). PMID: 25006137, pp. 1123–1131. DOI: 10.1377/hlthaff.2014.0041. eprint: https://doi.org/10.1377/hlthaff.2014.0041. URL: https://doi.org/10.1377/hlthaff.2014.0041.

[4] Idar Johan Brekke et al. "The value of vital sign trends in predicting and monitoring clinical deterioration: A systematic review". In: *PLOS ONE* 14 (Jan. 2019), e0210875. DOI: 10.1371/journal.pone.0210875.

[5] Chao Ma et al. "EDDI: Efficient Dynamic Discovery of High-Value Information with Partial VAE". In: *CoRR* abs/1809.11142 (2018). arXiv: 1809.11142. URL: http://arxiv.org/abs/1809.11142.

[6] Sean Barnes et al. "Real-time prediction of inpatient length of stay for discharge prioritization". In: *J Am Med Inform Assoc* 23.e1 (Apr. 2016). ocv106[PII], e2–e10. ISSN: 1527-974X. DOI: 10.1093/jamia/ocv106. URL: https://www.ncbi.nlm.nih.gov/pubmed/26253131.

[7] Majogé Vliet, Martijn Huisman, and Dorly J. H. Deeg. "Decreasing Hospital Length of Stay: Effects on Daily Functioning in Older Adults". In: *Journal of the American Geriatrics Society* 65.6 (June 2017), pp. 1214–1221. ISSN: 0002-8614. DOI: 10.1111/jgs.14767. URL: https://doi.org/10.1111/jgs.14767.

[8] Ayman Alahmar, Emad Mohammed, and Rachid Benlamri. "Application of Data Mining Techniques to Predict the Length of Stay of Hospitalized Patients with Diabetes". In: Aug. 2018, pp. 38–43. DOI: 10.1109/Innovate-Data.2018.00013.

[9] Kathleen Carey. "Hospital Length of Stay and Cost: A Multilevel Modeling Analysis". In: *Health Services and Outcomes Research Methodology* 3.1 (Mar. 2002), pp. 41–56. ISSN: 1572-9400. DOI: 10.1023/A:1021530924455. URL: https://doi.org/10.1023/A:1021530924455.

[10] Symeon Missios and Kimon Bekelis. "Drivers of hospitalization cost after craniotomy for tumor resection: creation and validation of a predictive model". In: *BMC Health Services Research* 15.1 (Mar. 2015), p. 85. ISSN: 1472-6963. DOI: 10.1186/s12913-015-0742-2. URL: https://doi.org/10.1186/s12913-015-0742-2.

[11] M. M. Stecker, M. Stecker, and J. Falotico. "Predictive model of length of stay and discharge destination in neuroscience admissions". In: *Surg Neurol Int* 8 (Feb. 2017). SNI-8-17[PII], pp. 17–17. ISSN: 2229-5097. DOI: 10.4103/2152-7806.199558. URL: https://www.ncbi.nlm.nih.gov/pubmed/28217396.

[12] Adriana Perez, Wenyaw Chan, and Rodolfo J. Dennis. "Predicting the Length of Stay of Patients Admitted for Intensive Care Using a First Step Analysis". In: *Health Serv Outcomes Res Methodol* 6.3-4 (Dec. 2006). PMC1828134[pmcid], pp. 127–138. ISSN: 1387-3741. DOI: 10.1007/s10742-006-0009-9. URL: https://www.ncbi.nlm.nih.gov/pubmed/18059977.

[13] A. Morton et al. "A Comparison of Supervised Machine Learning Techniques for Predicting Short-Term In-Hospital Length of Stay among Diabetic Patients". In: *2014 13th International Conference on Machine Learning and Applications.* Dec. 2014, pp. 428–431. DOI: 10.1109/ICMLA.2014.76.

[14] Pei-Fang Jennifer Tsai et al. "Length of Hospital Stay Prediction at the Admission Stage for Cardiology Patients Using Artificial Neural Network". In: *J Healthc Eng* 2016 (2016). 7035463[PII], p. 7035463. ISSN: 2040-2295. DOI: 10.1155/2016/7035463. URL: https://www.ncbi.nlm.nih.gov/pubmed/27195660.

[15] Whitney E Muhlestein et al. "Predicting Inpatient Length of Stay After Brain Tumor Surgery: Developing Machine Learning Ensembles to Improve Predictive Performance". In: (Aug. 2018). DOI: 10.1093/neuros/nyy343. eprint: http://oup.prod.sis.lan/neurosurgery/advance-article-pdf/doi/10.1093/neuros/nyy343/25423079/nyy343.pdf. URL: https://dx.doi.org/10.1093/neuros/nyy343.

[16] Peyman Rezaei Hachesu et al. "Use of data mining techniques to determine and predict length of stay of cardiac patients". In: *Healthc Inform Res* 19.2 (June 2013). PMC3717435[pmcid], pp. 121–129. ISSN: 2093-3681. DOI: 10.4258/hir.2013.19.2.121. URL: https://www.ncbi.nlm.nih.gov/pubmed/23882417.

[17] Astrid Schneider, Gerhard Hommel, and Maria Blettner. "Linear regression analysis: part 14 of a series on evaluation of scientific publications". In: *Dtsch Arztebl Int* 107.44 (Nov. 2010). PMC2992018[pmcid], pp. 776–782. ISSN: 1866-0452. DOI: 10.3238/arztebl.2010.0776. URL: https://www.ncbi.nlm.nih.gov/pubmed/21116397.

[18] Sandro Sperandei. "Understanding logistic regression analysis". In: *Biochem Med (Zagreb)* 24.1 (Feb. 2014). biochem-24-1-12-4[PII], pp. 12–18. ISSN: 1330-0962. URL: https://www.ncbi.nlm.nih.gov/pubmed/24627710.

[19] W S Ho, S Y Ying, and David Andrew Ross Burd. "Outcome analysis of 286 severely burned patients: retrospective study". In: *Hong Kong medical journal = Xianggang yi xue za zhi / Hong Kong Academy of Medicine* 8 (Sept. 2002), pp. 235–9.

[20] L. Rokach and O. Maimon. "Top-down induction of decision trees classifiers - a survey". In: *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 35.4 (Nov. 2005), pp. 476–487. ISSN: 1094-6977. DOI: 10.1109/TSMCC.2004.843247.

[21] Yan-Yan Song and Ying Lu. "Decision tree methods: applications for classification and prediction". In: *Shanghai Arch Psychiatry* 27.2 (Apr. 2015). sap-27-02-130[PII], pp. 130–135. ISSN: 1002-0829. URL: https://www.ncbi.nlm.nih.gov/pubmed/26120265.

[22] Maciej A. Mazurowski et al. "Training neural network classifiers for medical decision making: the effects of imbalanced datasets on classification performance". In: *Neural Netw* 21.2-3 (2008). S0893-6080(07)00240-7[PII], pp. 427–436. ISSN: 0893-6080. DOI: 10.1016/j.neunet.2007.12.031. URL: https://www.ncbi.nlm.nih.gov/pubmed/18272329.

[23] Corinna Cortes and Vladimir Vapnik. "Support-vector networks". In: *Machine Learning* 20.3 (Sept. 1995), pp. 273–297. DOI: 10.1007/BF00994018. URL: https://doi.org/10.1007/BF00994018.

[24] Christopher J.C. Burges. "A Tutorial on Support Vector Machines for Pattern Recognition". In: *Data Mining and Knowledge Discovery* 2.2 (June 1998), pp. 121–167. ISSN: 1573-756X. DOI: 10.1023/A:1009715923555. URL: https://doi.org/10.1023/A:1009715923555.

[25] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. "U-Net: Convolutional Networks for Biomedical Image Segmentation". In: *CoRR* abs/1505.04597 (2015). arXiv: 1505.04597. URL: http://arxiv.org/abs/1505.04597.

[26] Alberto Garcia-Garcia et al. "A Review on Deep Learning Techniques Applied to Semantic Segmentation". In: *CoRR* abs/1704.06857 (2017). arXiv: 1704.06857. URL: http://arxiv.org/abs/1704.06857.

[27] Alvin Rajkomar et al. “Scalable and accurate deep learning with electronic health records”. In: *npj Digital Medicine* 1.1 (2018), p. 18. ISSN: 2398-6352. DOI: 10.1038/s41746-018-0029-1. URL: https://doi.org/10.1038/s41746-018-0029-1.

[28] Leo Breiman. “Bagging predictors”. In: *Machine Learning* 24.2 (Aug. 1996), pp. 123–140. ISSN: 1573-0565. DOI: 10.1007/BF00058655. URL: https://doi.org/10.1007/BF00058655.

[29] M. Tsai et al. “16 Forecasting Medical Patient Length of Stay at Presentation in an Emergency Department Using Machine Learning”. In: *Annals of Emergency Medicine* 72.4 (Oct. 2018), S8. ISSN: 0196-0644. DOI: 10.1016/j.annemergmed.2018.08.021. URL: https://doi.org/10.1016/j.annemergmed.2018.08.021.

[30] Yoav Freund and Robert E Schapire. “A Short Introduction to Boosting”. In: *Journal of Japanese Society for Artificial Intelligence* 14 (Oct. 1999), pp. 771–780.

[31] S Arndt et al. “E-046 Length of stay in mechanical thrombectomy, and machine learning improvement of predictive analysis”. In: *Journal of NeuroInterventional Surgery* 9.Suppl 1 (2017), A64–A64. ISSN: 1759-8478. DOI: 10.1136/neurintsurg-2017-SNIS.118. eprint: https://jnis.bmj.com/content/9/Suppl_1/A64.1.full.pdf. URL: https://jnis.bmj.com/content/9/Suppl_1/A64.1.

[32] Chin-Sheng Yang et al. “Predicting the length of hospital stay of burn patients: Comparisons of prediction accuracy among different clinical stages”. In: *Decision Support Systems* 50.1 (2010), pp. 325–335. ISSN: 0167-9236. DOI: https://doi.org/10.1016/j.dss.2010.09.001. URL: http://www.sciencedirect.com/science/article/pii/S0167923610001703.

[33] Ruxandra Stoean et al. “Ensemble of Classifiers for Length of Stay Prediction in Colorectal Cancer”. In: *Advances in Computational Intelligence*. Ed. by Ignacio Rojas, Gonzalo Joya, and Andreu Catala. Cham: Springer International Publishing, 2015, pp. 444–457. ISBN: 978-3-319-19258-1.

TRITA CBH-GRU-2019:081