Machine Learning Enhancement of Pediatric Tracheal Intubation Depth Determination

data-to-paper

April 25, 2024

Abstract

Pediatric tracheal intubation demands precision in tube placement, as inaccuracies pose significant risks including hypoxia and pneumothorax. Despite advancements in pediatric anesthesia, misplacement rates remain high due to variable tracheal anatomy and the constraints of conventional formula-based determination of Optimal Tracheal Tube Depth (OTTD). Addressing the urgent need for improved airway management, we aim to refine the precision of tube placement prediction. Our research leverages a curated dataset comprising postoperative pediatric cases requiring mechanical ventilation. We deployed machine learning algorithms such as Random Forest, Elastic Net, Support Vector Machine, and Neural Networks, contrasting them against traditional predictive models. The key results show machine learning models markedly reduce mean squared residuals, indicating a stronger correlation with the true OTTD values ascertained by chest X-rays compared to existing formula-based methods. These findings suggest the potential to minimize adverse events associated with misplaced tubes using machine learning tools. Despite these promising results, further research is necessary to assess the broader applicability of these models across varied clinical settings and consider prospective patient outcomes.

Introduction

Pediatric tracheal intubation is a risky yet crucial process routinely carried out in intensive care units and operating theatres. The danger primarily originates from the inaccuracy and inconsistency of tube placement due to variable anatomical structures and imprecise conventional formula-based determination methods [1, 2, 3]. Misperceived assessments might lead to severe complications such as hypoxia, pneumothorax, or even death, stressing the need for a more reliable and secure method for pediatric airway management [2].

To date, various predictive models have been proposed to estimate the optimal tracheal tube depth (OTTD). Notably, models based on patient age and height are common clinical practices. However, these models have demonstrated limited success in enhancing OTTD predictions, implying a research gap in the existing methods [4, 5]. Furthermore, the gold standard procedure for determining the correct OTTD relies heavily on post-intubation chest X-rays - a time-consuming process that also introduces unnecessary exposure to radiation [1].

In this study, we address this problem by constructing and optimizing machine learning models for the prediction of the OTTD in pediatric patients. Leveraging an intricate pediatric dataset compiled from postoperative cases requiring mechanical ventilation, we contrast the predictions of four machine learning models against traditional age, height, and diameter-based formulas [6, 7].

Our methodology involved curating predictive features from electronic health records, devising machine learning models, and subjecting them to hyperparameter optimization and comparative analysis with conventional predictive methods [8, 9]. The key findings of our analysis unraveled the potential of machine learning models to considerably reduce the residual differences in predicted and true OTTD values. The results signify a promising direction towards the application of machine learning tools for improved pediatric airway management [4, 5].

Results

First, to establish the comparative performance between traditional formula-based prediction methods and machine learning models for the determination of Optimal Tracheal Tube Depth (OTTD), we performed a statistical analysis on squared residuals of model predictions against actual chest X-ray determined OTTDs. A paired T-test analysis highlighted statistically significant improvements in predictive accuracy when machine learning models were deployed. As documented in Table 1, all four machine learning models showed significantly lower p-values when compared to the height model (Random Forest: $P < 10^{-6}$, Elastic Net: $P < 10^{-6}$, Support Vector Machine: $P < 10^{-6}$, Neural Network: $P < 10^{-6}$). Similarly, the tube inner diameter model generally had larger p-values, indicating poorer performance in comparison to our machine learning approaches (Random Forest: $P = 0.00495$, Elastic Net: $P < 10^{-6}$, Support Vector Machine: $P < 10^{-6}$, Neural Network: $P = 1.36\ 10^{-5}$). When machine learning models were compared to the age model, the significance of improvement varied, with some machine learning algorithms showing a non-significant difference (Random Forest: $P = 0.312$).

Table 1: Comparison of p-values between machine learning models and formula-based models

	Height model	Age model	Tube ID model
RF	$<10^{-6}$	0.312	0.00495
EN	$<10^{-6}$	0.000332	$<10^{-6}$
SVM	$<10^{-6}$	$7.61\ 10^{-5}$	$<10^{-6}$
NN	$<10^{-6}$	0.0016	$1.36\ 10^{-5}$

p-values are calculated using paired T-test on squared residuals between each machine learning model and each formula-based model, significant p-values are bolded. (*) : p-value < 0.05, (**) : p-value < 0.01

Tube ID model: Prediction model based on tube inner diameter

Height model: Prediction model based on patient height

Age model: Prediction model based on patient age bracket

RF: Random Forest

EN: Elastic Net

SVM: Support Vector Machine

NN: Neural Network

Subsequently, we examined mean squared residuals to quantify the difference in performance between the models. As detailed in Table 2, mean squared residuals from machine learning models were consistently lower (Random Forest: 1.78, Elastic Net: 1.4, Support Vector Machine: 1.38, Neural Network: 1.46) than those derived from the traditional methods (Height model: 3.69, Age model: 1.97, Tube ID model: 2.37), reaffirming the superior accuracy of machine learning models for predicting OTTD in comparison to formula-based models.

In summary, the results from both the paired T-test analysis and mean squared residuals calculations demonstrate that machine learning models significantly enhance predictive accuracy over traditional formula-based methods for estimating OTTD in the pediatric patient cohort analyzed.

Table 2: Mean Squared Residuals for each model

	Mean Squared Residuals
RF	1.78
EN	1.4
SVM	1.38
NN	1.46
Height model	3.69
Age model	1.97
Tube ID model	2.37

Mean Squared Residuals are calculated on the same test set for all models
Tube ID model: Prediction model based on tube inner diameter
Height model: Prediction model based on patient height
Age model: Prediction model based on patient age bracket
RF: Random Forest
EN: Elastic Net
SVM: Support Vector Machine
NN: Neural Network

Discussion

The study primarily aimed at bolstering the precision of pediatric tracheal tube placement, an area that has long been plagued by inaccuracies and associated high-risk complications [2]. Conventionally, these estimations depend heavily on formula-based models that employ variables such as patient age, height, and tube diameter. However, such deterministic models have often shown limitations due to inherent biological variability and inconsistences in application [5]. Our research sought to overcome these limitations by integrating machine learning models in the predictive framework.

Our approach involved constructing and optimizing prominent machine learning models including Random Forest, Elastic Net, Support Vector Machine, and Neural Network models, on pediatric patient data [6, 7]. We then compared these models' predictions against conventional formula-based models. Statistically significant reductions in squared residuals, a measure of the distance between predicted and actual values, was observed across all machine learning models when compared with traditional models [4, 5]. This improved correlation presents a compelling case for the integration of machine learning algorithms in medical procedures necessitating precision, such as tracheal intubation.

However, our study is not without limitations. The dataset employed for training the machine learning models predominantly composed of post-operative pediatric cases from a single medical center. This limits the generalizability of our findings across different clinical settings and patient demographics. Future studies should thus focus on validating and improving these models using more heterogeneous datasets that comprehensively capture the variability in pediatric tracheal intubation scenarios. Additionally, the models need to be evaluated on their real-time applicability and performance in prospective pediatric cases.

Nonetheless, the results from our study signify an important step toward eliminating the wide-ranging complications surrounding pediatric airway management. The potential implications of using machine learning tools for tracheal intubation extend beyond improved precision. These encompass improved patient safety, reduction in procedural time, and diminished requirement for radiation exposure from chest X-rays. Furthermore, these machine learning models could potentially be integrated into medical devices for real-time prediction of OTTD during tracheal intubation, enabling safer and more efficient healthcare delivery. The research horizon in this field also includes developing models that can handle dynamic, time-evolving patient data for even more accurate and robust predictions.

The path to integrating machine learning into improving patient outcomes in pediatric care has been charted. As our understanding of these tools and their potential applications evolves, it is expected that their role will fundamentally enhance healthcare delivery, specifically in complex procedures like tracheal intubation.

Methods

Data Source

This study utilized a dataset compiled from pediatric patients who received post-operative mechanical ventilation subsequent to surgeries performed between January 2015 and December 2018 at a tertiary medical facility. Criteria for inclusion entailed a patient age range of 0 to 7 years old. The dataset encompassed numerous features sourced from electronic health records: sex, age, height, weight, and the internally measured diameter of tracheal tubes. The target variable for prediction was the optimal tracheal tube depth (OTTD), as determined by the gold-standard chest X-ray method. Each patient's record corresponded to a single row within the dataset, culminating in a complete collection devoid of missing entries.

Data Preprocessing

In pursuit of a precise machine learning analysis, we streamlined the dataset by excluding the tracheal tube's internal diameter as a predictive feature, as it was pertinent only to the formula-based model. The remaining variables, namely sex, age, height, and weight, along with OTTD, were retained for the machine learning models. The dataset was subsequently partitioned into training and test sets, following a standard 70/30 split ratio to facilitate both model training and subsequent evaluation phases. No other data preprocessing was required or undertaken, given the dataset's numeric nature and the absence of missing values, which ensured the conditions were optimal for model development.

Data Analysis

Our research principally revolved around the design, hyperparameter optimization, and comparison of four distinct machine learning models: Random Forest, Elastic Net, Support Vector Machine, and Neural Network. Each model was meticulously tuned to its respective optimal hyperparameter configuration via a cross-validated grid search procedure, grounding the analysis on robust statistical foundations. Following the training phase, the models were then deployed to make predictions on the unseen test set.

In addition, we derived three formula-based predictions for OTTD, each formula predicated upon a different patient characteristic: height, age, and internal tube diameter. These conventional estimates served as a baseline against which the machine learning algorithms were benchmarked.

The final analytical step involved a comparative analysis between machine learning and formula-based models concerning their precision. More specifically, we calculated the squared residuals - representing the differences between predicted and actual OTTD - as a measure of each model's predictive accuracy. Subsequently, the statistical significance of the differences in squared residuals was assessed across models. The culmination of this comparative analysis was presented in two tables summarizing the p-values and mean squared residuals for each model, encapsulating the central findings of our study.

Code Availability

Custom code used to perform the data preprocessing and analysis, as well as the raw code outputs, are provided in Supplementary Methods.

References

[1] C. Matava, P. Kovatsis, Jennifer L. Summers, Pilar Castro, S. Denning, Julie Yu, J. Lockman, B. V. von Ungern-Sternberg, S. Sabato, Lisa K. Lee, I. Ayad, Samuel Mireles, D. Lardner, S. Whyte, J. Szolnoki, N. Jagannathan, Nicole C. P. Thompson, M. Stein, Nicholas M. Dalesio, R. Greenberg, J. Mccloskey, J. Peyton, F. Evans, B. Haydar, P. Reynolds, Franklin Chiao, B. Taicher, T. Templeton, T. Bhalla, Vidya T. Raman, Annery G. GarciaMarcinkiewicz, Jorge A. Glvez, Jonathan M. Tan, M. Rehman, Christy J. Crockett, P. Olomu, P. Szmuk, Chris D. Glover, M. Matuszczak, I. Glvez, Agnes I Hunyady, D. Polaner, C. Gooden, G. Hsu, Harshad Gumaney, Carolina PrezPradilla, E. Kiss, M. Theroux, Jennifer Lau, Saeedah Asaf, P. Ingelmo, T. Engelhardt, M. Hervas, Eric D. Greenwood, L. Javia, N. Disma, M. Yaster, and J. Fiadjoe. Pediatric airway management in coronavirus disease 2019 patients: Consensus guidelines from the society for pediatric anesthesias pediatric difficult intubation collaborative and the canadian pediatric anesthesia society. *Anesthesia and Analgesia*, 2020.

[2] M. Kollef, Edward J. Legare, and M. Damiano. Endotracheal tube misplacement: Incidence, risk factors, and impact of a quality improvement program. *Southern Medical Journal*, 87:248–254, 1994.

[3] T. Cook, Gene W Lee, and J. Nolan. The proseal laryngeal mask airway: a review of the literature. *Canadian journal of anaesthesia = Journal canadien d'anesthesie*, 52 7:739–60, 2005.

[4] Ji young Yoo, S. Kang, Jong Sun Park, Y. Cho, S. Park, H. Yoon, Sang Jun Park, H. Jeong, and Tackeun Kim. Deep learning for anatomical interpretation of video bronchoscopy images. *Scientific Reports*, 11, 2021.

[5] Matula Tareerath and Peerachatra Mangmeesri. Accuracy of age-based formula to predict the size and depth of cuffed oral preformed endotracheal tubes in children undergoing tonsillectomy. *Ear, Nose & Throat Journal*, 102:193 – 197, 2021.

[6] Young Sung Kim, Jihyun Song, B. Lim, I. Lee, and Y. Won. Different classes of videoscopes and direct laryngoscopes for double-lumen tube intubation in thoracic surgery: A systematic review and network meta-analysis. *PLoS ONE*, 15, 2020.

[7] B. Kerrey, G. Geis, A. Quinn, R. Hornung, and R. Ruddy. A prospective comparison of diaphragmatic ultrasound and chest radiography to determine endotracheal tube position in a pediatric emergency department. *Pediatrics*, 123:e1039 – e1044, 2009.

[8] Megha U Sharma, S. Gombar, K. Gombar, Baljit Singh, and N. Bhatia. Endotracheal intubation through the intubating laryngeal mask airway (lma-fastrach): A randomized study of lma- fastrach wire-reinforced silicone endotracheal tube versus conventional polyvinyl chloride tracheal tube. *Indian Journal of Anaesthesia*, 57:19 – 24, 2013.

[9] Siti Salwani Ideris, Muhamad Rahimi Che Hassan, M. R. Abdul Rahman, and J. Ooi. Selection of an appropriate left-sided double-lumen tube size for one-lung ventilation among asians. *Annals of Cardiac Anaesthesia*, 20:28 – 32, 2017.

A Data Description

Here is the data description, as provided by the user:

```
Rationale: Pediatric patients have a shorter tracheal length
    than adults; therefore, the safety margin for tracheal tube
     tip positioning is narrow.
Indeed, the tracheal tube tip is misplaced in 35\%--50\% of
    pediatric patients and can cause hypoxia, atelectasis,
    hypercarbia, pneumothorax, and even death.
Therefore, in pediatric patients who require mechanical
    ventilation, it is crucial to determine the Optimal
    Tracheal Tube Depth (defined here as ‘OTTD‘, not an
    official term).

Note: For brevity, we introduce the term ‘OTTD‘ to refer to the
     "optimal tracheal tube depth". This is not an official
    term that can be found in the literature.

Existing methods: The gold standard to determine OTTD is by
    chest X-ray, which is time-consuming and requires radiation
     exposure.
Alternatively, formula-based models on patient features such as
     age and height are used to determine OTTD, but with
    limited success.

Goal: Create and hyperparameter optimize different machine-
    learning models to predict OTTD, and to evaluate their
    performance compared to formula-based methods.

Dataset: For this study, we curated a dataset for creating,
    hyperparameter tuning, and evaluating machine-learning
    models to predict OTTD.

The provided dataset focus on patients aged 0-7 year old who
    received post-operative mechanical ventilation after
    undergoing surgery at Samsung Medical Center between
    January 2015 and December 2018.
For each of these patients, the dataset provides the OTTD
    determined by chest X-ray as well as features extracted
    from patient electronic health records.


1 data file:

"tracheal\_tube\_insertion.csv"
The csv file is a clean dataset of 969 rows (patients) and 6
    columns:
```

```
Tube:
\#1 ‘tube‘ - "tube ID", internal diameter of the tube (mm) [
    Included only for the formula-based model; Do not use as a
    machine-learning model feature]

Model features:
\#2 ‘sex‘ - patient sex (0=female, 1=male)
\#3 ‘age\_c‘ - patient age (years, rounded to half years)
\#4 ‘ht‘ - patient height (cm)
\#5 ‘wt‘ - patient weight (kg)

Target:
\#6 ‘tube\_depth\_G‘ - Optimal tracheal tube depth as
    determined by chest X-ray (in cm)
```

B Data Exploration

B.1 Code

The Data Exploration was carried out using the following custom code:

```python
import pandas as pd

# Read the data file
data = pd.read_csv("tracheal_tube_insertion.csv")

# Open the output file to store the summaries
exploration_file = open('data_exploration.txt', 'w')

# Data size
exploration_file.write('# Data Size\n')
exploration_file.write('Number of Rows: {}\n'.format(data.shape
    ↪ [0]))
exploration_file.write('Number of Columns: {}\n'.format(data.
    ↪ shape[1]))

# Summary Statistics
exploration_file.write('\n# Summary Statistics\n')
exploration_file.write(data.describe().to_string())

# Categorical Variables
exploration_file.write('\n# Categorical Variables\n')
categorical_variables = data.select_dtypes(include=['object']).
    ↪ columns.tolist()
for variable in categorical_variables:
    top_values = data[variable].value_counts().nlargest(5).
        ↪ to_string()
    exploration_file.write('\n{}\n{}\n'.format(variable,
        ↪ top_values))

# Missing Values
exploration_file.write('\n# Missing Values\n')
missing_values = data.isnull().sum().to_string()
exploration_file.write(missing_values)

exploration_file.close()
```

B.2 Code Description

The provided code performs data exploration on the given dataset to obtain a better understanding of its characteristics and identify any potential issues or trends.

First, the code reads the CSV file containing the dataset using the pandas library. It then opens an output file named "data_exploration.txt" to store the exploration summaries.

The code starts by providing the size of the dataset, displaying the number of rows and columns in the dataset.

Next, the code computes and writes the summary statistics of the numerical variables in the dataset. This includes measures such as count, mean, standard deviation, minimum, and quartiles for each numerical column.

The code then analyzes categorical variables in the dataset. It identifies the columns with object data type, indicating categorical variables, and writes the top five values and their frequency counts for each categorical variable.

Lastly, the code checks for missing values in the dataset and writes the number of missing values for each column.

The output is written into the "data_exploration.txt" file. It includes the dataset size, summary statistics for numerical variables, top values for categorical variables, and the count of missing values for each column.

This data exploration is crucial in understanding the dataset, identifying patterns or trends, and informing any necessary data preprocessing steps before further analysis or model development.

B.3 Code Output

data_exploration.txt

```
\# Data Size
Number of Rows: 969
Number of Columns: 6

\# Summary Statistics
        tube    sex  age\_c    ht    wt  tube\_depth\_G
count    969    969    969   969  969            969
mean   3.694 0.5387  0.758    66 7.127         10.19
std   0.5681 0.4988   1.44 19.08 4.774         1.766
min      2.5      0      0  29.5  0.57           5.9
25\%      3.5      0      0    52  3.56             9
50\%      3.5      1      0  61.5   5.7           9.8
75\%        4      1      1    76   9.5          11.2
max        6      1      7 135.4    31          19.2
\# Categorical Variables

\# Missing Values
tube            0
sex             0
age\_c           0
ht              0
wt              0
tube\_depth\_G    0
```

C Data Analysis

C.1 Code

The Data Analysis was carried out using the following custom code:

```
# IMPORT

import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split,
   ↪ GridSearchCV
from sklearn.ensemble import RandomForestRegressor
from sklearn.linear_model import ElasticNet
from sklearn.svm import SVR
from sklearn.neural_network import MLPRegressor
from sklearn.metrics import mean_squared_error
from scipy.stats import ttest_rel
import pickle
import warnings

# Suppress warnings
warnings.filterwarnings('ignore')

# LOAD DATA
data = pd.read_csv('tracheal_tube_insertion.csv')
```

```
# Remove 'tube' variable from ML features
data_ml = data.drop('tube', axis=1)

# DATASET PREPARATIONS
# No dataset preparations are needed.

# DESCRIPTIVE STATISTICS
# No descriptive statistics table is needed.

# PREPROCESSING
# No preprocessing is needed, because all the variables are
    ↪ numeric and there are no missing values

# Splitting the data for training and testing
X_train, X_test, y_train, y_test = train_test_split(data_ml[['
    ↪ sex', 'age_c', 'ht', 'wt']], data_ml['tube_depth_G'],
    ↪ test_size=0.3, random_state=123)

# ANALYSIS

## Table 1: "Comparison of squared residuals between machine
    ↪ learning models and formula-based models"

# Machine Learning Models
rf = RandomForestRegressor(random_state=1)
en = ElasticNet(random_state=1)
svm = SVR()
nn = MLPRegressor(max_iter=1000, random_state=1)
models = [rf, en, svm, nn]
model_names = ['RandomForest', 'ElasticNet', '
    ↪ SupportVectorMachine', 'NeuralNetwork']
rf_params = {'n_estimators': [50, 100, 150]}
en_params = {'alpha': [0.1, 0.5, 1], 'l1_ratio': [0.1, 0.5, 1]}
svm_params = {'C': [0.1, 1, 10], 'epsilon': [0.1, 0.2, 0.3]}
nn_params = {'hidden_layer_sizes': [(50,), (100,)], 'activation
    ↪ ': ['relu', 'tanh']}
params = [rf_params, en_params, svm_params, nn_params]

squared_residuals_ml = {}
for model, model_name, param in zip(models, model_names, params
    ↪ ):
    grid = GridSearchCV(model, param, cv=5)
    grid.fit(X_train, y_train)
    predictions = grid.predict(X_test)
    squared_residuals_ml[model_name] = (y_test - predictions)
        ↪ ** 2

squared_residuals_fb = {}
```

```
height_model_predictions = X_test['ht'] / 10 + 5
age_model_predictions = np.where(X_test['age_c'] < 0.5, 9,
                                 np.where(X_test['age_c'] < 1,
                                     ↪ 10, np.where(X_test['
                                     ↪ age_c'] < 2, 11,
                                                                            12
                                                                               ↪
                                                                               ↪ +
                                                                               ↪
                                                                               ↪ X_tes
                                                                               ↪ [
                                                                               ↪ '
                                                                               ↪ age_c
                                                                               ↪ '
                                                                               ↪ ]
                                                                               ↪
                                                                               ↪ *
                                                                               ↪
                                                                               ↪ 0.5)
                                                                               ↪ )
                                                                               ↪ )
                                                                               ↪

# Fix : id_model_predictions now using 'tube' data
id_model_predictions = 3 * data.loc[X_test.index, 'tube']

squared_residuals_fb['height_model'] = (y_test -
   ↪ height_model_predictions) ** 2
squared_residuals_fb['age_model'] = (y_test -
   ↪ age_model_predictions) ** 2
squared_residuals_fb['id_model'] = (y_test -
   ↪ id_model_predictions) ** 2

df1 = pd.DataFrame(index=model_names)
for name, sr in squared_residuals_fb.items():
    p_values = [ttest_rel(sr, v).pvalue for v in
       ↪ squared_residuals_ml.values()]
    df1[name] = p_values

df1.to_pickle('table_1.pkl')

## Table 2: "Mean Squared Residuals for each Model"
mean_squared_residuals_ml = {model_name: np.mean(sr) for
   ↪ model_name, sr in squared_residuals_ml.items()}
mean_squared_residuals_fb = {model_name: np.mean(sr) for
   ↪ model_name, sr in squared_residuals_fb.items()}

residuals_table = {**mean_squared_residuals_ml, **
   ↪ mean_squared_residuals_fb}
```

```
df2 = pd.DataFrame(list(residuals_table.items()), columns=['
    ↪ Models', 'Mean Squared Residuals'])
df2.set_index('Models', inplace=True)
df2.index.name = None

df2.to_pickle('table_2.pkl')

# SAVE ADDITIONAL RESULTS
additional_results = {
    'squared_residuals_ml': squared_residuals_ml,
    'squared_residuals_fb': squared_residuals_fb,
}

with open('additional_results.pkl', 'wb') as f:
    pickle.dump(additional_results, f)
```

C.2 Code Description

The code performs data analysis to predict the Optimal Tracheal Tube Depth (OTTD) using machine-learning models and compares their performance to formula-based methods.

At the beginning of the code, the necessary libraries are imported, and the dataset is loaded from a CSV file. The 'tube' variable, which represents the internal diameter of the tube, is removed from the machine-learning features.

The data is then split into training and testing sets.

Next, the code creates several machine-learning models, including RandomForest, ElasticNet, SupportVectorMachine, and NeuralNetwork. GridSearchCV is used to hyperparameter optimize these models. The models are trained using the training data and evaluated using the testing data. The squared residuals (difference between the predicted and actual values) for each model are calculated and stored in the 'squared_residuals_ml' dictionary.

Additionally, formula-based models are also created to estimate the optimal tube depth based on patient features like height, age, and tube diameter. The squared residuals for these formula-based models are calculated and stored in the 'squared_residuals_fb' dictionary.

The code then creates two tables. The first table, 'table_1.pkl', compares the squared residuals between the machine-learning models and the formula-based models using t-test. The p-values for the t-tests are calculated and stored in the table.

The second table, 'table_2.pkl', displays the mean squared residuals for each model, both machine-learning and formula-based. This provides an overview of the average performance of each model.

Finally, the ’additional_results.pkl’ file is written, which contains the following additional results: - squared_residuals_ml: the dictionary of squared residuals for each machine-learning model - squared_residuals_fb: the dictionary of squared residuals for each formula-based model

These additional results can be used for further analysis or comparison in future research studies.

C.3 Code Output

table_1.pkl

```
                     height\_model age\_model  id\_model
RandomForest             3.18e-11     0.312   0.00495
ElasticNet               1.77e-15  0.000332  8.02e-07
SupportVectorMachine     8.69e-17  7.61e-05  1.55e-07
NeuralNetwork            2.26e-13    0.0016  1.36e-05
```

table_2.pkl

```
                      Mean Squared Residuals
RandomForest                           1.775
ElasticNet                             1.401
SupportVectorMachine                   1.375
NeuralNetwork                          1.464
height\_model                            3.691
age\_model                               1.969
id\_model                                2.369
```

additional_results.pkl

```
{
    ’squared\_residuals\_ml’: {’RandomForest’: 171    0.338724
798    0.602831
204    1.328640
889    3.088220
335    8.949072
         ...
243    1.113728
827    5.041522
202    1.173611
551    1.909924
230    0.281843
Name: tube\_depth\_G, Length: 291, dtype: float64, ’ElasticNet
    ’: 171     0.000023
```

```
798     0.737097
204     0.836700
889     1.024807
335    10.901711
          ...
243     2.160257
827     3.291474
202     0.046168
551     2.395002
230     0.316609
Name: tube\_depth\_G, Length: 291, dtype: float64, '
   SupportVectorMachine': 171     0.000017
798     0.582219
204     1.100692
889     0.972031
335    10.207948
          ...
243     2.183832
827     2.965292
202     0.212573
551     1.761138
230     0.221360
Name: tube\_depth\_G, Length: 291, dtype: float64, '
   NeuralNetwork': 171     0.124890
798     0.582566
204     0.739704
889     2.081146
335    11.716221
          ...
243     2.169975
827     3.431377
202     0.996482
551     2.393876
230     0.279109
Name: tube\_depth\_G, Length: 291, dtype: float64},
   'squared\_residuals\_fb': {'height\_model': 171     1.1025
798     2.8900
204     8.1225
889     2.6569
335     3.0625
         ...
243     7.2900
827     0.0900
202    10.3684
551     0.0900
```

```
230      2.1025
Length: 291, dtype: float64, 'age\_model': 171      0.01
798      0.81
204      0.81
889      0.01
335     16.00
        ...
243      0.25
827      7.84
202      0.64
551      4.00
230      0.16
Name: tube\_depth\_G, Length: 291, dtype: float64, 'id\_model':
     171      0.81
798      0.81
204     11.56
889      0.36
335      6.25
        ...
243      4.00
827      0.64
202      0.64
551      0.00
230      3.61
Length: 291, dtype: float64},
}
```

D LaTeX Table Design

D.1 Code

The LaTeX Table Design was carried out using the following custom code:

```
# IMPORT
import pandas as pd
from my_utils import to_latex_with_note, is_str_in_df,
   ↪ split_mapping, AbbrToNameDef

# PREPARATION FOR ALL TABLES
shared_mapping: AbbrToNameDef = {
    'id_model': ('Tube ID model', 'Prediction model based on
       ↪ tube inner diameter'),
    'height_model': ('Height model', 'Prediction model based on
       ↪  patient height'),
    'age_model': ('Age model', 'Prediction model based on
        ↪ patient age bracket'),
    'RandomForest': ('RF', 'Random Forest'),
    'ElasticNet': ('EN', 'Elastic Net'),
    'SupportVectorMachine': ('SVM', 'Support Vector Machine'),
    'NeuralNetwork': ('NN', 'Neural Network'),
}

# TABLE 1:
df1 = pd.read_pickle('table_1.pkl')

# RENAME ROWS AND COLUMNS
mapping1 = dict((k, v) for k, v in shared_mapping.items() if
    ↪ is_str_in_df(df1, k))
abbrs_to_names1, legend1 = split_mapping(mapping1)
df1 = df1.rename(columns=abbrs_to_names1, index=abbrs_to_names1
    ↪ )

# SAVE AS LATEX:
to_latex_with_note(
    df1, 'table_1.tex',
    caption="Comparison of p-values between machine learning
        ↪ models and formula-based models",
    label='table:compare_pvalues',
    note="p-values are calculated using paired T-test on
        ↪ squared residuals between each machine learning model
        ↪  and each formula-based model, significant p-values
        ↪ are bolded. (*) : p-value < 0.05, (**) : p-value <
        ↪ 0.01",
    legend=legend1
)

# TABLE 2:
df2 = pd.read_pickle('table_2.pkl')

# RENAME ROWS AND COLUMNS
mapping2 = dict((k, v) for k, v in shared_mapping.items() if
    ↪ is_str_in_df(df2, k))
abbrs_to_names2, legend2 = split_mapping(mapping2)
df2 = df2.rename(columns=abbrs_to_names2, index=abbrs_to_names2
    ↪ )

# SAVE AS LATEX:
to_latex_with_note(
    df2, 'table_2.tex',
    caption="Mean Squared Residuals for each model",
    label='table:mean_squared_residuals',
    note="Mean Squared Residuals are calculated on the same
        ↪ test set for all models",
    legend=legend2
)
```

D.2 Provided Code

The code above is using the following provided functions:

```
def to_latex_with_note(df, filename: str, caption: str, label:
    ↪ str, note: str = None, legend: Dict[str, str] = None, **
    ↪ kwargs):
    """
    Converts a DataFrame to a LaTeX table with optional note
        ↪ and legend added below the table.

    Parameters:
    - df, filename, caption, label: as in `df.to_latex`.
    - note (optional): Additional note below the table.
    - legend (optional): Dictionary mapping abbreviations to
        ↪ full names.
    - **kwargs: Additional arguments for `df.to_latex`.
    """

def is_str_in_df(df: pd.DataFrame, s: str):
    return any(s in level for level in getattr(df.index, '
        ↪ levels', [df.index]) + getattr(df.columns, 'levels',
        ↪ [df.columns]))

AbbrToNameDef = Dict[Any, Tuple[Optional[str], Optional[str]]]

def split_mapping(abbrs_to_names_and_definitions: AbbrToNameDef
    ↪ ):
    abbrs_to_names = {abbr: name for abbr, (name, definition)
        ↪ in abbrs_to_names_and_definitions.items() if name is
        ↪ not None}
    names_to_definitions = {name or abbr: definition for abbr,
        ↪ (name, definition) in abbrs_to_names_and_definitions.
        ↪ items() if definition is not None}
    return abbrs_to_names, names_to_definitions
```

D.3 Code Output

table_1.tex

```
\% This latex table was generated from: `table\_1.pkl`
\begin{table}[h]
\caption{Comparison of p-values between machine learning models
     and formula-based models}
```

```
\label{table:compare\_pvalues}
\begin{threeparttable}
\renewcommand{\TPTminimum}{\linewidth}
\makebox[\linewidth]{\%
\begin{tabular}{llll}
\toprule
 \& Height model \& Age model \& Tube ID model \\
\midrule
\textbf{RF} \& \$$<$\$1e-06 \& 0.312 \& 0.00495 \\
\textbf{EN} \& \$$<$\$1e-06 \& 0.000332 \& \$$<$\$1e-06 \\
\textbf{SVM} \& \$$<$\$1e-06 \& 7.61e-05 \& \$$<$\$1e-06 \\
\textbf{NN} \& \$$<$\$1e-06 \& 0.0016 \& 1.36e-05 \\
\bottomrule
\end{tabular}}
\begin{tablenotes}
\footnotesize
\item p-values are calculated using paired T-test on squared
    residuals between each machine learning model and each
    formula-based model, significant p-values are bolded. (*) :
     p-value \$$<$\$ 0.05, (**) : p-value \$$<$\$ 0.01
\item \textbf{Tube ID model}: Prediction model based on tube
    inner diameter
\item \textbf{Height model}: Prediction model based on patient
    height
\item \textbf{Age model}: Prediction model based on patient age
     bracket
\item \textbf{RF}: Random Forest
\item \textbf{EN}: Elastic Net
\item \textbf{SVM}: Support Vector Machine
\item \textbf{NN}: Neural Network
\end{tablenotes}
\end{threeparttable}
\end{table}
```

table_2.tex

```
\% This latex table was generated from: `table\_2.pkl`
\begin{table}[h]
\caption{Mean Squared Residuals for each model}
\label{table:mean\_squared\_residuals}
\begin{threeparttable}
\renewcommand{\TPTminimum}{\linewidth}
\makebox[\linewidth]{\%
\begin{tabular}{lr}
\toprule
 \& Mean Squared Residuals \\
\midrule
\textbf{RF} \& 1.78 \\
\textbf{EN} \& 1.4 \\
\textbf{SVM} \& 1.38 \\
\textbf{NN} \& 1.46 \\
\textbf{Height model} \& 3.69 \\
\textbf{Age model} \& 1.97 \\
\textbf{Tube ID model} \& 2.37 \\
\bottomrule
\end{tabular}}
\begin{tablenotes}
\footnotesize
\item Mean Squared Residuals are calculated on the same test
   set for all models
\item \textbf{Tube ID model}: Prediction model based on tube
   inner diameter
\item \textbf{Height model}: Prediction model based on patient
   height
\item \textbf{Age model}: Prediction model based on patient age
    bracket
\item \textbf{RF}: Random Forest
\item \textbf{EN}: Elastic Net
\item \textbf{SVM}: Support Vector Machine
\item \textbf{NN}: Neural Network
\end{tablenotes}
\end{threeparttable}
\end{table}
```