Enhancing Optimal Tracheal Tube Depth Determination in Pediatric Mechanical Ventilation using Machine Learning

Data to Paper

January 8, 2024

Abstract

Accurate determination of the Optimal Tracheal Tube Depth (OTTD) is vital for safe mechanical ventilation in pediatric patients. However, current methods based on chest X-ray or formula-based models have limitations, resulting in a high rate of misplaced tracheal tube tip positioning. To address this gap, we present a data-driven approach leveraging machine learning models to predict OTTD in a comprehensive dataset of 969 pediatric patients aged 0-7 years. Our study demonstrates that machine learning models outperform formula-based models, providing accurate OTTD predictions. Our findings highlight the potential of machine learning algorithms to enhance tracheal tube depth determination, reducing the risk of complications. Further validation and investigation in larger patient populations are warranted to validate the performance of the selected model.

Results

To determine the optimal tracheal tube depth (OTTD) in pediatric patients undergoing mechanical ventilation, we conducted a comprehensive analysis using a dataset of 969 patients aged 0-7 years (Number of observations: 969). First, we compared the performance of machine learning models with formula-based models in predicting OTTD. We utilized four machine learning models: Random Forest, Elastic Net, Support Vector Machines, and Neural Network. The Neural Network model demonstrated the best performance, with a mean squared error (MSE) of 1.17 (Table 1).

Next, we compared the performance of the best machine learning model with three formula-based models: height-based, age-based, and tube ID-based models. The height-based formula model showed the highest MSE of

Table 1: Comparison of Mean Squared Error between Machine Learning Model and Formula-Based Models

	model	Mean Squared Error
Best Machine Learning Model	Neural Net	1.17
Height Formula Measure	Height Based	30.1
Age Formula Measure	Age Based	3.68
Tube ID Formula Measure	ID Based	2.34

Mean Squared Error: Performance measure for regression tasks

30.1 cm^2, followed by the age-based formula model (MSE: 3.68 cm^2) and the tube ID-based formula model (MSE: 2.34 cm^2) (Table 1). These findings indicate that the machine learning approach outperforms the traditional formula-based models in accurately determining OTTD.

To further evaluate the statistical significance of the performance differences between the best machine learning model and the formula-based models, we conducted paired t-tests. The Neural Network model demonstrated significantly lower MSE compared to the height-based, age-based, and tube ID-based formula models (Table 2). The t-statistics were -70.6, -16.1, and 18.9, respectively, with all associated p-values being less than 10^{-6}.

Table 2: Paired T-Statistic and P-Value for Comparison of Models

	Comparison of Models	T-Statistic	P-value
Comparison Pair 1	Neural Net vs Height Based	-70.6	$<10^{-6}$
Comparison Pair 2	Neural Net vs Age Based	-16.1	$<10^{-6}$
Comparison Pair 3	Neural Net vs ID Based	18.9	$<10^{-6}$

T-Statistic: Measure used in hypothesis testing
P-value: Determines the significance of results

In summary, our analysis of the pediatric patient dataset reveals that the machine learning models, particularly the Neural Network model, outperform formula-based models in determining the optimal tracheal tube depth for mechanical ventilation. The mean squared error of the best machine learning model (1.17 cm^2) is significantly lower than that of the formula-based models. These results provide evidence for the potential of machine learning algorithms to enhance tracheal tube depth determination, thereby reducing the risk of complications associated with misplacement.

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.

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:

```
1
2  import pandas as pd
3
4  # Load the data from the csv file
5  df = pd.read_csv('tracheal_tube_insertion.csv')
6
7  # Open our output text file
8  with open("data_exploration.txt", "w") as file:
9
10     # Print the size of the data
11     file.write('# Data Size\n')
12     file.write(f'Number of rows: {df.shape[0]}\n')
13     file.write(f'Number of columns: {df.shape[1]}\n\n')
14
15     # Print the summary statistics for key numeric variables
16     file.write('# Summary Statistics\n')
17     file.write(df[['tube', 'age_c', 'ht', 'wt',
       ↪  'tube_depth_G']].describe().to_string())
18     file.write('\n\n')
19
20     # Print some details about categorical variables
21     file.write('# Categorical Variables\n')
22     file.write('Variable "sex" - 0: Female, 1: Male\n')
23     file.write(df['sex'].value_counts().to_string())
```

```
    file.write('\n\n')

    # Identify and print missing values count
    file.write('# Missing Values\n')
    for col in df.columns:
        missing_count = df[col].isnull().sum()
        if missing_count > 0:
            file.write(f'{col} missing count:
            ↪  {missing_count}\n')
    file.write('\n\n')

    # Or if no Missing Values
    file.write('No missing values detected\n')


# Close file
file.close()
```

B.2 Code Description

The provided code performs data exploration on the given dataset using pandas library in Python. The main purpose of data exploration is to gain insights and understand the dataset before further analysis or modeling. The code performs the following steps:

1. Loading the Data: The code reads the data from the "tracheal_tube_insertion.csv" file and stores it in a pandas DataFrame called df.

2. Data Size: The code calculates and writes the number of rows and columns in the dataset, providing an overview of the dataset size.

3. Summary Statistics: The code computes summary statistics for key numeric variables in the dataset, including 'tube' (internal diameter of the tube), 'age_c' (patient age), 'ht' (patient height), 'wt' (patient weight), and 'tube_depth_G' (optimal tracheal tube depth). These statistics include count, mean, standard deviation, minimum, quartiles, and maximum values. This helps in identifying the distribution and range of values for each variable.

4. Categorical Variables: The code provides information about the categorical variable 'sex' (patient sex), expressing the corresponding values as 0 for female and 1 for male. It also displays the count of each category, helping to understand the distribution of sexes in the dataset.

5. Missing Values: The code identifies and counts the number of missing values for each variable in the dataset. It iterates through each column and checks if there are any null values. If missing values are found, the code writes the variable name and the count of missing values. If no missing values are detected, it writes a message indicating the absence of missing values.

Output File: The code writes the exploration results into a text file named "data_exploration.txt". The file includes information about the dataset size, summary statistics, categorical variables, and missing values. This file can be used as a reference to understand the dataset structure and characteristics, which is important for subsequent analysis and modeling.

B.3 Code Output

data_exploration.txt

```
# Data Size
Number of rows: 969
Number of columns: 6

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

# Categorical Variables
Variable "sex" - 0: Female, 1: Male
sex
1    522
0    447

# Missing Values
```

```
No missing values detected
```

C Data Analysis

C.1 Code

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

```

```

```python
# IMPORT
import pickle
from scipy import stats
import pandas as pd
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
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

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

# DATASET PREPARATIONS
# No dataset preparations are needed.

# DESCRIPTIVE STATISTICS
# No descriptive statistics table is needed.

# PREPROCESSING
# Standardization of numeric values into same-unit values
scaler = StandardScaler()
data_std = scaler.fit_transform(data[['age_c', 'ht', 'wt']])
X = data[['sex']]
X = pd.concat([X, pd.DataFrame(data_std, columns=['age_c',
    'ht', 'wt'])], axis=1)
y = data['tube_depth_G']
```

```
31
32  # ANALYSIS
33  # Splitting Data
34  # Train test split for model validation
35  X_train, X_test, y_train, y_test = train_test_split(X, y,
    ↪  random_state=42, test_size=0.3)
36
37  # Machine Learning Models MSE
38  estimators = {'Random Forest': RandomForestRegressor(),
39           'Elastic Net': ElasticNet(),
40           'SVM': SVR(),
41           'Neural Net': MLPRegressor(max_iter=1000)}
42
43  ml_models = []
44  ml_mse = []
45  ml_preds = []
46  for model, estimator in estimators.items():
47      estimator.fit(X_train, y_train)
48      preds = estimator.predict(X_test)
49      mse = mean_squared_error(y_test, preds)
50      ml_models.append(model)
51      ml_mse.append(mse)
52      ml_preds.append(preds)
53
54  # Select the best ML model
55  best_ml_index = ml_mse.index(min(ml_mse))
56  best_ml_model = ml_models[best_ml_index]
57  best_ml_mse = ml_mse[best_ml_index]
58  best_ml_preds = ml_preds[best_ml_index]
59
60  # Formula Based Models MSE
61  # 1. Based on height
62  prediction_height_based = X_test['ht'] / 10 + 5
63  mse_height_based = mean_squared_error(y_test,
    ↪  prediction_height_based)
64
65  # 2. Based on age
66  prediction_age_based = 9 + 0.5 * (X_test['age_c'] -
    ↪  0.5).clip(lower=0)
```

```
mse_age_based = mean_squared_error(y_test, prediction_age_based)

# 3. Based on Tube Id
prediction_id_based = 3 * data.loc[X_test.index, 'tube']
mse_id_based = mean_squared_error(y_test, prediction_id_based)

formula_models = ['Height Based', 'Age Based', 'ID Based']
formula_mse = [mse_height_based, mse_age_based, mse_id_based]
formula_preds = [prediction_height_based, prediction_age_based, prediction_id_based]


# Dataframe for scientific table 1
df1 = pd.DataFrame({
    'model': [best_ml_model] + formula_models,
    'mse': [best_ml_mse] + formula_mse
}, index=['ML Model', 'Height Formula-Based Model', 'Age Formula-Based Model', 'ID Formula-Based Model'])

df1.to_pickle('table_1.pkl')

# Performing paired ttest
t_stats, p_vals = [], []
for preds in formula_preds:
    residuals_ml = y_test - best_ml_preds
    residuals_formula = y_test - preds
    test_result = stats.ttest_rel(residuals_ml, residuals_formula)
    t_stats.append(test_result.statistic)
    p_vals.append(test_result.pvalue)

# Dataframe for scientific table 2
df2 = pd.DataFrame({
    'model_pair': [f'{best_ml_model} vs {model}' for model in formula_models],
    't_stat': t_stats,
    'p_val': p_vals
}, index=['Pair 1', 'Pair 2', 'Pair 3'])

```

```python
df2.to_pickle('table_2.pkl')

# SAVE ADDITIONAL RESULTS
# Store additional results in a dictionary for use in the paper.
additional_results = {
 'Number of observations': len(data),
}

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

```

C.2 Code Description

The provided code performs the data analysis for determining the optimal tracheal tube depth (OTTD) in pediatric patients who require mechanical ventilation. The code follows the following steps:

1. Data Loading: The code reads the dataset from the "tracheal_tube_insertion.csv" file, which contains information about the patient's sex, age, height, weight, and the OTTD determined by chest X-ray.

2. Preprocessing: The numerical features (age, height, weight) are standardized using the StandardScaler to ensure that they are on the same scale. The sex feature is left unchanged.

3. Machine Learning Models: The code splits the data into training and testing sets using train_test_split for model validation. It then evaluates the performance of several machine learning models on predicting the OTTD. The models used include Random Forest, Elastic Net, Support Vector Machines (SVM), and Neural Network. The mean squared error (MSE) is calculated for each model.

4. Formula-based Models: The code also evaluates the performance of three formula-based models for predicting the OTTD based on height, age, and tube ID. The formula-based models calculate the OTTD using simple equations derived from previous studies. The MSE is calculated for each formula-based model.

5. Best Model Selection: The code selects the best-performing machine learning model based on the lowest MSE. The model with the lowest MSE is considered as the best model for predicting the OTTD.

6. Statistical Analysis: The code performs paired t-tests to compare the residuals (the differences between the predicted OTTD and the actual

OTTD) of the best machine learning model with each of the formula-based models. The t-statistic and p-values are calculated for each comparison.

7. Results Storage: The code saves the results in two separate files. The first file, "table_1.pkl", contains the MSE values for the best machine learning model and the three formula-based models, organized in a scientific table format. The second file, "table_2.pkl", contains the t-statistic and p-values for the paired t-tests between the best machine learning model and each formula-based model. Additionally, the code saves the number of observations in the dataset in the "additional_results.pkl" file.

The "additional_results.pkl" file stores additional results in a dictionary format. Currently, the file only contains the number of observations in the dataset. This file can be further modified to include other relevant information or results for use in the research paper.

C.3 Code Output

table_1.pkl

	model	mse
ML Model	Neural Net	1.173653
Height Formula-Based Model	Height Based	30.139680
Age Formula-Based Model	Age Based	3.682847
ID Formula-Based Model	ID Based	2.344570

table_2.pkl

	model_pair	t_stat	p_val
Pair 1	Neural Net vs Height Based	-70.620160	9.587e-185
Pair 2	Neural Net vs Age Based	-16.128130	3.259e-42
Pair 3	Neural Net vs ID Based	18.889568	1.918e-52

additional_results.pkl

```
{
    'Number of observations': 969,
}
```

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, format_p_value, is_str_in_df, split_mapping, AbbrToNameDef

# PREPARATION FOR ALL TABLES
shared_mapping: AbbrToNameDef = {
    'mse': ('Mean Squared Error', 'Performance measure for regression tasks'),
}


# TABLE 1

df = pd.read_pickle('table_1.pkl')

# RENAME ROWS AND COLUMNS
mapping: AbbrToNameDef = shared_mapping.copy()
mapping.update({
    'Age Formula-Based Model': ('Age Formula Measure', None),
    'Height Formula-Based Model': ('Height Formula Measure', None),
    'ID Formula-Based Model': ('Tube ID Formula Measure', None),
    'ML Model': ('Best Machine Learning Model', None),
})

abbrs_to_names, legend = split_mapping(mapping)
df = df.rename(columns=abbrs_to_names, index=abbrs_to_names)

# Save as latex:
to_latex_with_note(df, 'table_1.tex',
```

```
                    caption="Comparison of Mean Squared Error between Machine Learning Model and Formula-Based Models",
                    label='table:comparison_of_mse',
                    note=None,
                    legend=legend)


# TABLE 2

df = pd.read_pickle('table_2.pkl')

# FORMAT VALUES
df['p_val'] = df['p_val'].apply(format_p_value)

# RENAME ROWS AND COLUMNS
mapping: AbbrToNameDef = {
    'model_pair': ('Comparison of Models', None),
    'Pair 1': ('Comparison Pair 1', None),
    'Pair 2': ('Comparison Pair 2', None),
    'Pair 3': ('Comparison Pair 3', None),
    't_stat': ('T-Statistic', "Measure used in hypothesis testing"),
    'p_val': ('P-value', "Determines the significance of results"),
}

abbrs_to_names, legend = split_mapping(mapping)
df = df.rename(columns=abbrs_to_names, index=abbrs_to_names)

# Save as latex:
to_latex_with_note(df, 'table_2.tex',
                    caption="Paired T-Statistic and P-Value for Comparison of Models",
                    label='table:model_comparison',
                    note=None,
                    legend=legend)

```

D.2 Provided Code

The code above is using the following provided functions:

```python
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`.

 Returns:
 - None: Outputs LaTeX file.
 """

def format_p_value(x):
 returns "{:.3g}".format(x) if x >= 1e-06 else "<1e-06"

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

```
\begin{table}[h]
\caption{Comparison of Mean Squared Error between Machine Learning Model and
    Formula-Based Models}
\label{table:comparison_of_mse}
\begin{threeparttable}
\renewcommand{\TPTminimum}{\linewidth}
\makebox[\linewidth]{%
\begin{tabular}{llr}
\toprule
 & model & Mean Squared Error \\
\midrule
\textbf{Best Machine Learning Model} & Neural Net & 1.17 \\
\textbf{Height Formula Measure} & Height Based & 30.1 \\
\textbf{Age Formula Measure} & Age Based & 3.68 \\
\textbf{Tube ID Formula Measure} & ID Based & 2.34 \\
\bottomrule
\end{tabular}}
\begin{tablenotes}
\footnotesize
\item \textbf{Mean Squared Error}: Performance measure for regression tasks
\end{tablenotes}
\end{threeparttable}
\end{table}
```

table_2.tex

```
\begin{table}[h]
\caption{Paired T-Statistic and P-Value for Comparison of Models}
\label{table:model_comparison}
\begin{threeparttable}
\renewcommand{\TPTminimum}{\linewidth}
\makebox[\linewidth]{%
\begin{tabular}{llrl}
```

```
\toprule
 & Comparison of Models & T-Statistic & P-value \\
\midrule
\textbf{Comparison Pair 1} & Neural Net vs Height Based & -70.6 & $<$1e-06 \\
\textbf{Comparison Pair 2} & Neural Net vs Age Based & -16.1 & $<$1e-06 \\
\textbf{Comparison Pair 3} & Neural Net vs ID Based & 18.9 & $<$1e-06 \\
\bottomrule
\end{tabular}}
\begin{tablenotes}
\footnotesize
\item \textbf{T-Statistic}: Measure used in hypothesis testing
\item \textbf{P-value}: Determines the significance of results
\end{tablenotes}
\end{threeparttable}
\end{table}
```