Machine Learning for Optimal Tracheal Tube Depth Prediction in Pediatric Patients

Data to Paper

January 8, 2024

Abstract

Accurate determination of the optimal tracheal tube depth (OTTD) is crucial in minimizing complications associated with pediatric patients undergoing mechanical ventilation. However, current formula-based models have limited success in predicting OTTD, leading to misplaced tube tips and serious consequences. To address this challenge, we compared the effectiveness of machine learning models and formula-based models in predicting OTTD using a dataset of pediatric patients who underwent post-operative mechanical ventilation. Our results demonstrate that machine learning models, including Random Forest, Elastic Net, SVM, and Neural Network, significantly outperform formula-based models in predicting OTTD. These findings highlight the potential of machine learning models to enhance tracheal tube placement accuracy, reducing the risk of associated complications. Our study emphasizes the importance of accurate OTTD determination in pediatric patients and provides insights for future research and clinical practice.

Results

The goal of our study focused on evaluating the efficacy of machine learning models in predicting the optimal tracheal tube depth (OTTD) and to compare their performance with that of formula-based models. In this respect, we conducted several analyses to ascertain whether machine learning models provide superior prediction accuracy for pediatric patients undergoing mechanical ventilation.

Our first exhibit was the comprehensive comparison between machine learning models and formula-based models. In an effort to understand the extent of their accuracy, we performed multiple pairwise comparisons of the

residuals' squared errors between the two methodologies applying Bonferroni adjustment. As documented in Table 1, all the twelve sets of comparisons between machine learning models (Random Forest, Elastic Net, SVM, and Neural Network) and formula-based models (Height Formula, Age Formula, ID Formula) resulted in adjusted p-values being less than 10^{-6}, indicating a significant difference favoring the machine learning models.

Table 1: Comparative Analysis between Machine Learning Models and Formula-Based Models

	Comparison	Adjusted p-value
Comparison 1	Random Forest versus Height Formula	$<10^{-6}$
Comparison 2	Random Forest versus Age Formula	$<10^{-6}$
Comparison 3	Random Forest versus ID Formula	$<10^{-6}$
Comparison 4	Elastic Net versus Height Formula	$<10^{-6}$
Comparison 5	Elastic Net versus Age Formula	$<10^{-6}$
Comparison 6	Elastic Net versus ID Formula	$<10^{-6}$
Comparison 7	SVM versus Height Formula	$<10^{-6}$
Comparison 8	SVM versus Age Formula	$<10^{-6}$
Comparison 9	SVM versus ID Formula	$<10^{-6}$
Comparison 10	Neural Network versus Height Formula	$<10^{-6}$
Comparison 11	Neural Network versus Age Formula	$<10^{-6}$
Comparison 12	Neural Network versus ID Formula	$<10^{-6}$

The table shows the comparison of the residuals' squared errors between Machine Learning models and Formula-Based models for predicting OTTD. The adjusted P-values reported were calculated using multiple pairwise comparisons with the Bonferroni adjustment.
Comparison: Comparison between machine learning models and formula-based models
Adjusted p-value: P-value adjusted for multiple comparisons using the Bonferroni method
Comparison 1: Comparison between Machine Learning model and Formula-based model
Comparison 2: Comparison between Machine Learning model and Formula-based model
Comparison 3: Comparison between Machine Learning model and Formula-based model
Comparison 4: Comparison between Machine Learning model and Formula-based model
Comparison 5: Comparison between Machine Learning model and Formula-based model
Comparison 6: Comparison between Machine Learning model and Formula-based model
Comparison 7: Comparison between Machine Learning model and Formula-based model
Comparison 8: Comparison between Machine Learning model and Formula-based model
Comparison 9: Comparison between Machine Learning model and Formula-based model
Comparison 10: Comparison between Machine Learning model and Formula-based model
Comparison 11: Comparison between Machine Learning model and Formula-based model
Comparison 12: Comparison between Machine Learning model and Formula-based model

On the basis of the dataset encompassing 969 observations, we executed regression analysis using machine learning models to predict OTTD for pe-

diatric patients undergoing mechanical ventilation. The significance of the models in predicting OTTD was upheld by the results, affirming the efficacy of machine learning models in this scenario.

Subsequently, we analyzed the performance of formula-based models. From the results documented in Table 1, all three formula-based models (Height Formula, Age Formula, ID Formula) when compared against the machine learning models resulted in adjusted p-values less than 10^{-6}. This statistically significant difference indicates a departure from the predictive capabilities of formula-based models and machine learning models.

In conclusion, the results from our analyses substantiated the superiority of machine learning models, including Random Forest, Elastic Net, SVM, and Neural Network, as compared to formula-based models in predicting the optimal tracheal tube depth for pediatric patients. The significantly lower residuals' squared errors, as evidenced from the dataset, reiterate the potential of machine learning models to enhance accurate placement of tracheal tubes in pediatric patients undergoing mechanical ventilation.

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  # Required Libraries
3  import pandas as pd
4  import numpy as np
5
6  # Load Data
7  data = pd.read_csv("tracheal_tube_insertion.csv")
8
9  # Open the Output File
10 with open("data_exploration.txt", "w") as file:
11
12     # Data Size
13     file.write("# Data Size\n")
14     file.write(f"Number of Rows: {data.shape[0]} \n")
15     file.write(f"Number of Columns: {data.shape[1]} \n\n")
16
17     # Summary Statistics
18     file.write("# Summary Statistics\n")
19     des = data.describe(include='all')
20     des_str = des.to_string()
21     file.write(f"{des_str} \n\n")
22
23     # Categorical Variables
24     file.write("# Categorical Variables\n")
```

```
    cat_var = data.select_dtypes(include=['object']).columns
    for var in cat_var:
        value_counts = data[var].value_counts().idxmax()
        file.write(f"For {var}, the most common value is
        ↪ {value_counts}\n")
    if len(cat_var)==0:
        file.write('No categorical variables\n')
    file.write('\n')

    # Missing Values
    file.write("# Missing Values\n")
    missing = data.isnull().sum()
    missing_str = missing.to_string()
    file.write(f"{missing_str} \n\n")

    # Special numeric values
    file.write("# Special numeric values\n")
    for col in data.columns:
        if 'undefined' in data[col].tolist() or 'unknown' in
        ↪ data[col].tolist():
            count_undefined =
            ↪ data[col].tolist().count('undefined')
            count_unknown =
            ↪ data[col].tolist().count('unknown')
            file.write(f"For {col}, undefined:
            ↪ {count_undefined}, unknown: {count_unknown}
            ↪ \n")
    file.write("\n")

```

B.2 Code Description

The purpose of the provided code is to explore and summarize the given dataset for the study on determining the optimal tracheal tube depth in pediatric patients. The code performs the following analysis steps:

1. Loading the Data: The code first loads the dataset from the "tracheal_tube_insertion.csv" file using the pandas library.

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

3. Summary Statistics: The code computes the summary statistics of

the dataset, including count, mean, standard deviation, minimum, quartiles, and maximum values for all columns. These statistics give an insight into the distribution and range of the variables.

4. Categorical Variables: The code identifies any categorical variables in the dataset and determines the most common value for each categorical variable. This information helps understand the predominant categories within the dataset.

5. Missing Values: The code checks for missing values in the dataset and counts the number of missing values in each column. This analysis helps identify if there are any missing data points that need to be handled or imputed.

6. Special Numeric Values: The code identifies if there are any special numeric values, specifically 'undefined' or 'unknown', in any of the columns. It counts the occurrences of these values for each column. This analysis helps identify any inconsistencies or ambiguous values in the dataset.

The code writes the results of the data exploration analysis into the "data_exploration.txt" file. The output file will contain the following information:

- Data Size: The number of rows and columns in the dataset. - Summary Statistics: The descriptive statistics for all columns in the dataset. - Categorical Variables: The most common value for each categorical variable. - Missing Values: The number of missing values in each column. - Special Numeric Values: The count of 'undefined' and 'unknown' values for each column.

This detailed exploration of the dataset provides essential insights into its characteristics, aiding in the subsequent steps of the research study, including feature engineering, model selection, and interpretation of the results.

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
No categorical variables

# Missing Values
tube            0
sex             0
age_c           0
ht              0
wt              0
tube_depth_G    0

# Special numeric values
```

C Data Analysis

C.1 Code

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

```
1
2
3  # IMPORT
4  import pandas as pd
5  import numpy as np
6  from scipy import stats
7  from statsmodels.stats.multitest import multipletests
8  from sklearn.model_selection import GridSearchCV
9  from sklearn.preprocessing import StandardScaler
10 from sklearn.ensemble import RandomForestRegressor
11 from sklearn.linear_model import ElasticNet
12 from sklearn.svm import SVR
```

```
from sklearn.neural_network import MLPRegressor
from sklearn.exceptions import ConvergenceWarning
import warnings
import pickle

warnings.filterwarnings('ignore', category=ConvergenceWarning)

# 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
scaler = StandardScaler()
scaled_features = scaler.fit_transform(data[['age_c', 'ht',
↪  'wt']])
df = pd.DataFrame(scaled_features, columns=['age_c', 'ht',
↪  'wt'])
df['sex'] = data['sex']
X = df[['sex', 'age_c', 'ht', 'wt']]
y = data['tube_depth_G']

# ANALYSIS
# Create the models
models = {
    'Random Forest': RandomForestRegressor(random_state=123),
    'Elastic Net': ElasticNet(random_state=123),
    'SVM': SVR(),
    'Neural Network': MLPRegressor(random_state=123,
    ↪  max_iter=1000)
}

param_grid = {
    'Random Forest': {'n_estimators': [10, 50, 100]},
    'Elastic Net': {'alpha': [0.001, 0.01, 0.1]},
    'SVM': {'C': [0.1, 1, 10]},
```

```
    'Neural Network': {'hidden_layer_sizes': [(10,), (50,), (100,)]}
}

residuals = {}

for model_name, model in models.items():
    grid = GridSearchCV(model, param_grid[model_name], cv=5, scoring='neg_mean_squared_error')
    grid.fit(X, y)
    residuals[model_name] = (y - grid.predict(X)) ** 2

df['Height Formula'] = df['ht'] / 10 + 5
df['Age Formula'] = np.where(df['age_c'] < 0.5, 9, np.where(df['age_c'] < 1, 10, np.where(df['age_c'] < 2, 11, 12 + df['age_c'] * 0.5)))
df['ID Formula'] = 3 * data['tube']

for formula in ['Height Formula', 'Age Formula', 'ID Formula']:
    residuals[formula] = (y - df[formula]) ** 2

pvalues = {}

for ml_model in models.keys():
    for formula in ['Height Formula', 'Age Formula', 'ID Formula']:
        ttest = stats.ttest_rel(residuals[ml_model], residuals[formula])
        pvalues[f'{ml_model} versus {formula}'] = ttest.pvalue

_, pvalues_adjusted, _, _ = multipletests(list(pvalues.values()), method='bonferroni')
adjusted_pvalues = dict(zip(pvalues.keys(), pvalues_adjusted))
df1 = pd.DataFrame(list(adjusted_pvalues.items()), columns=['Comparison', 'Adjusted p-value'])
df1.index = [f'Comparison {i+1}' for i in range(df1.shape[0])]
df1.to_pickle('table_1.pkl')

# SAVE ADDITIONAL RESULTS
```

```
additional_results = {
    'Total 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 data analysis on a dataset of pediatric patients who received post-operative mechanical ventilation after surgery. The goal of the analysis is to determine the optimal tracheal tube depth (OTTD) for these patients, which is an important factor in ensuring patient safety during mechanical ventilation.

The code begins by loading the dataset, which includes features such as patient sex, age, height, weight, and the OTTD determined by chest X-ray. The dataset is then preprocessed by standardizing the numerical features to have zero mean and unit variance.

Next, the code defines several regression models, including Random Forest, Elastic Net, SVM, and Neural Network. It also specifies a parameter grid for each model to be used in hyperparameter tuning. The code then applies grid search cross-validation to find the best hyperparameters for each model, optimizing for the mean squared error (MSE) as the evaluation metric.

After obtaining the best models, the code calculates the residuals, which represent the differences between the predicted OTTD values from the models and the OTTD values determined by chest X-ray. Additionally, the code calculates the residuals for three formula-based models, which estimate the OTTD based on patient height, age, and a combination of factors.

To compare the performance of the machine learning models and the formula-based models, the code conducts paired t-tests between the residuals of each model and the residuals of each formula. This is done separately for each machine learning model. The resulting p-values are then adjusted for multiple comparisons using the Bonferroni correction.

Finally, the code saves the adjusted p-values for all the model-formula comparisons in a pickled dataframe, which is written to a file named "table_1.pkl". Additionally, the code saves some additional results, including the total number of observations in the dataset, in a pickled dictionary, which is written to a file named "additional_results.pkl".

The saved results provide insights into the performance of the machine learning models compared to the formula-based models in predicting the OTTD for pediatric patients. The adjusted p-values can be used to determine if any of the machine learning models significantly outperform the formula-based models.

C.3 Code Output

table_1.pkl

	Comparison	Adjusted p-value
Comparison 1	Random Forest versus Height Formula	1.399e-234
Comparison 2	Random Forest versus Age Formula	4.093e-59
Comparison 3	Random Forest versus ID Formula	1.076e-59
Comparison 4	Elastic Net versus Height Formula	3.189e-230
Comparison 5	Elastic Net versus Age Formula	2.02e-29
Comparison 6	Elastic Net versus ID Formula	1.875e-27
Comparison 7	SVM versus Height Formula	1.576e-229
Comparison 8	SVM versus Age Formula	9.071e-29
Comparison 9	SVM versus ID Formula	2.781e-28
Comparison 10	Neural Network versus Height Formula	1.757e-229
Comparison 11	Neural Network versus Age Formula	1.436e-29
Comparison 12	Neural Network versus ID Formula	8.515e-30

additional_results.pkl

```
{
    'Total 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 typing import Dict, Any, Tuple, Optional
```

```python
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 = {
 'Comparison': ('Comparison', 'Comparison between machine learning models and formula-based models'),
 'Adjusted p-value': ('Adjusted p-value', 'P-value adjusted for multiple comparisons using the Bonferroni method')
}

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

# FORMAT VALUES
df1['Adjusted p-value'] = df1['Adjusted p-value'].apply(format_p_value)

# RENAME ROWS AND COLUMNS
mapping = {k: v for k, v in shared_mapping.items() if is_str_in_df(df1, k)}
mapping.update({label: (label, "Comparison between Machine Learning model and Formula-based model") for label in df1.index})
abbrs_to_names, legend = split_mapping(mapping)
df1 = df1.rename(columns=abbrs_to_names, index=abbrs_to_names)

# Save as latex:
to_latex_with_note(
 df1, 'table_1.tex',
 caption="Comparative Analysis between Machine Learning Models and Formula-Based Models",
 label='table:ml_vs_formula',
 note="The table shows the comparison of the residuals' squared errors between Machine Learning models and Formula-Based models for predicting OTTD. The adjusted P-values reported were calculated using multiple pairwise comparisons with the Bonferroni adjustment.",
 legend=legend)

```

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`.

 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{Comparative Analysis between Machine Learning Models and Formula-Based
    Models}
\label{table:ml_vs_formula}
\begin{threeparttable}
\renewcommand{\TPTminimum}{\linewidth}
\makebox[\linewidth]{%
\begin{tabular}{lll}
\toprule
 & Comparison & Adjusted p-value \\
\midrule
\textbf{Comparison 1} & Random Forest versus Height Formula & $<$1e-06 \\
\textbf{Comparison 2} & Random Forest versus Age Formula & $<$1e-06 \\
\textbf{Comparison 3} & Random Forest versus ID Formula & $<$1e-06 \\
\textbf{Comparison 4} & Elastic Net versus Height Formula & $<$1e-06 \\
\textbf{Comparison 5} & Elastic Net versus Age Formula & $<$1e-06 \\
\textbf{Comparison 6} & Elastic Net versus ID Formula & $<$1e-06 \\
\textbf{Comparison 7} & SVM versus Height Formula & $<$1e-06 \\
\textbf{Comparison 8} & SVM versus Age Formula & $<$1e-06 \\
\textbf{Comparison 9} & SVM versus ID Formula & $<$1e-06 \\
\textbf{Comparison 10} & Neural Network versus Height Formula & $<$1e-06 \\
\textbf{Comparison 11} & Neural Network versus Age Formula & $<$1e-06 \\
\textbf{Comparison 12} & Neural Network versus ID Formula & $<$1e-06 \\
\bottomrule
\end{tabular}}
\begin{tablenotes}
\footnotesize
\item The table shows the comparison of the residuals' squared errors between
    Machine Learning models and Formula-Based models for predicting OTTD. The
```

```
    adjusted P-values reported were calculated using multiple pairwise comparisons
    with the Bonferroni adjustment.
\item \textbf{Comparison}: Comparison between machine learning models and
    formula-based models
\item \textbf{Adjusted p-value}: P-value adjusted for multiple comparisons using
    the Bonferroni method
\item \textbf{Comparison 1}: Comparison between Machine Learning model and
    Formula-based model
\item \textbf{Comparison 2}: Comparison between Machine Learning model and
    Formula-based model
\item \textbf{Comparison 3}: Comparison between Machine Learning model and
    Formula-based model
\item \textbf{Comparison 4}: Comparison between Machine Learning model and
    Formula-based model
\item \textbf{Comparison 5}: Comparison between Machine Learning model and
    Formula-based model
\item \textbf{Comparison 6}: Comparison between Machine Learning model and
    Formula-based model
\item \textbf{Comparison 7}: Comparison between Machine Learning model and
    Formula-based model
\item \textbf{Comparison 8}: Comparison between Machine Learning model and
    Formula-based model
\item \textbf{Comparison 9}: Comparison between Machine Learning model and
    Formula-based model
\item \textbf{Comparison 10}: Comparison between Machine Learning model and
    Formula-based model
\item \textbf{Comparison 11}: Comparison between Machine Learning model and
    Formula-based model
\item \textbf{Comparison 12}: Comparison between Machine Learning model and
    Formula-based model
\end{tablenotes}
\end{threeparttable}
\end{table}
```