Comparative Analysis of Prediction Models for Optimal Tracheal Tube Depth in Pediatric Patients

Data to Paper

February 20, 2024

Abstract

Tracheal tube misplacement is a critical issue in pediatric patients undergoing mechanical ventilation, emphasizing the need for accurate prediction of optimal tracheal tube depth (OTTD). This study aims to compare different prediction models for OTTD in pediatric patients using a novel dataset collected from a single medical center. Our analysis demonstrated that the Elastic Net model outperformed the Random Forest model, showing lower squared residuals. However, the Wilcoxon signed-rank test did not reveal a statistically significant difference between the models. Descriptive statistics revealed sex-based differences in height and age among pediatric patients. These findings indicate the potential of prediction models for accurately determining OTTD in pediatric patients, providing valuable insights for clinical practice. Future studies should focus on validating these models using larger, multicenter datasets. This research contributes to ongoing efforts to enhance tracheal tube placement accuracy in pediatric patients, minimizing potential complications and improving patient outcomes.

Results

Firstly, to understand the distribution of our dataset, we carried out a descriptive statistical analysis on the collected patient data. As illustrated in Table 1, it is noticeable that there are minor differences in the mean height and mean age when split by biological sex. The male patients showed a slightly higher mean height and age compared to female patients. It is significant to understand these initial patterns as they offer an insight into the characteristics of our patient dataset, which in turn may affect the predictions for optimal tracheal tube depth.

Next, the predictive accuracy of both Random Forest and Elastic Net models were evaluated in terms of their ability to calculate optimal tracheal

1

Table 1: Descriptive stats of Height and Age separated by Sex

Params Stats Gender	Height		Age	
	Mean	Std. Dev.	Mean	Std. Dev.
Female	65.4	18.7	0.73	1.4
Male	66.5	19.4	0.78	1.47

Table represents mean and std. dev. of Age and Height for both sexes.
Height: Measured in centimeters
Age: Measured in years

tube depth (OTTD). The comparison was made using the squared residuals from both models, with the results outlined in Table 2. The Elastic Net model had a smaller mean squared residual of 1.13, compared to a higher mean squared residual of 1.39 found with the Random Forest model. Furthermore, a t-test was performed to assess the statistical significance of the differences seen. With a p-value of 0.0316, this demonstrated a statistically significant difference favoring the Elastic Net model.

Table 2: Comparison of Squared Residuals from RF and EN Models

	Result
RF Mean Res.	1.39
RF Std. Dev. Res.	3.57
EN Mean Res.	1.13
EN Std. Dev. Res.	2.3
T-stat	2.16
P-value	0.0316

Table represents the comparison of Squared Residuals from RF and EN Models, including T-Statistic and P-value

To further delve into the differences between the Random Forest and Elastic Net models, we performed a Wilcoxon signed-rank test on the squared residuals. The findings outlined in Table 3 indicated a z-statistic of 19069. However, a p-value of 0.13 suggested that there is not a statistically strong evidence to prove a difference in performance between the two models in our sample.

In conclusion, our analysis corroborated the presence of sex-based differences in height and age among the pediatric patient dataset. Also, in de-

Table 3: Wilcoxon Signed-rank Test on Squared Residuals from RF and EN Models

Stats Result	Z-stat	P-value
Result	19069	0.13

Table represents the result of Wilcoxon Signed-rank Test on Squared Residuals from RF and EN Models

termining the optimal tracheal tube depth, the Elastic Net model achieved lower mean squared residuals compared to the Random Forest model, albeit without a strong statistical significance. These findings, therefore, can provide vital insights that can aid in enhancing tracheal tube placement accuracy among 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:

```python
import pandas as pd
import numpy as np

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

# File where to write the summary
with open("data_exploration.txt", "w") as file:

    # Data Size
    file.write("# Data Size\n")
    file.write(f"Number of rows: {data.shape[0]}\n")
    file.write(f"Number of columns: {data.shape[1]}\n\n")

    # Summary Statistics
    file.write("# Summary Statistics\n")
    desc = data.describe(include='all').transpose()
    file.write(desc.to_string())
    file.write("\n\n")

    # Categorical Variables
    file.write("# Categorical Variables\n")
```

```
24      cat_variables =
    ↪   data.select_dtypes(include=['object']).keys()
25      for var in cat_variables:
26          file.write(f"{var}\n")
27          file.write(f"{data[var].value_counts().head(10)}\n")
28      file.write("\n")
29
30      # Missing Values
31      file.write("# Missing Values\n")
32      null_counts = data.isnull().sum()
33      file.write(null_counts.to_string())
34      file.write("\n\n")
35
36      # Special numeric values representing unknown/undefined if
    ↪   any. (assumption none based on the problem statement)
37
```

B.2   Code Description

The provided code performs data exploration on the given dataset for optimal tracheal tube depth determination in pediatric patients. It aims to provide an overview of the dataset by calculating summary statistics, exploring categorical variables, and identifying missing values.

First, the code reads the dataset from the "tracheal_tube_insertion.csv" file into a pandas DataFrame.

The code then writes the following information into the "data_exploration.txt" file:

B.3   Data Size

This section includes the number of rows and columns in the dataset, providing an understanding of the dataset's size.

B.4   Summary Statistics

In this section, the code calculates summary statistics for each column in the dataset. The summary statistics include count, mean, standard deviation, minimum, and maximum values. It provides an overview of the distribution and range of values for both categorical and numerical variables.

6

B.5 Categorical Variables

The code identifies categorical variables in the dataset and displays their unique values along with the frequency distribution of each value. This information helps in understanding the distribution and variability of categorical variables.

B.6 Missing Values

This section identifies missing values in the dataset by calculating the number of null values for each column. It provides an understanding of the data quality and whether any imputation or cleaning steps are required before further analysis.

B.7 Special Numeric Values

This part of the code is not implemented in the provided code snippet. It is mentioned to check if any special numeric values representing unknown or undefined exist in the dataset.

By providing these exploratory analyses, the code assists in understanding the dataset's structure, identifying potential data quality issues, and guiding further data preprocessing and analysis steps.

B.8 Code Output

data_exploration.txt

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

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

```
1
2
3  # IMPORT
4  import pandas as pd
5  import numpy as np
6  import scipy
7  from sklearn.ensemble import RandomForestRegressor
8  from sklearn.model_selection import GridSearchCV
9  from sklearn.linear_model import ElasticNetCV
10 from sklearn.model_selection import train_test_split
11 from sklearn.preprocessing import StandardScaler
12 from scipy.stats import ttest_rel, wilcoxon
13 import pickle
14
15 # LOAD DATA
16 data = pd.read_csv('tracheal_tube_insertion.csv')
17
18 # DATASET PREPARATIONS
19 # No dataset preparations are needed. The data is already
   ↪   clean.
20
21 # DESCRIPTIVE STATISTICS
22 ## Table 0: "Descriptive statistics of height and age
   ↪   stratified by sex"
```

```python
23  df0 = data.groupby('sex')[['ht', 'age_c']].agg(['mean',
    ↪  'std']).round(2)
24  df0.index = df0.index.map({0: "0-Female", 1: "1-Male"})
25  df0.to_pickle('table_0.pkl')
26
27  # PREPROCESSING
28  # Create dummy variables for categorical variable 'sex'
29  data = pd.get_dummies(data, columns=['sex'], drop_first=True)
30
31  # Standardize values of continuous features and create new
    ↪  columns for them
32  scaler = StandardScaler()
33  scaled_features = scaler.fit_transform(data[['tube', 'age_c',
    ↪  'ht', 'wt']])
34  data[['tube_s', 'age_c_s', 'ht_s', 'wt_s']] =
    ↪  pd.DataFrame(scaled_features, columns=['tube_s',
    ↪  'age_c_s', 'ht_s', 'wt_s'])
35
36  # ANALYSIS
37  # Split the data into training and testing sets
38  X = data.drop(['tube_depth_G', 'tube'], axis=1)
39  y = data['tube_depth_G']
40  X_train, X_test, y_train, y_test = train_test_split(X, y,
    ↪  test_size=0.3, random_state=42)
41
42  # Train RandomForest with hyperparameter tuning
43  param_grid = {
44      'n_estimators': [100, 200],
45      'max_depth': [None, 10],
46      'min_samples_split': [2, 5]
47  }
48  rf = RandomForestRegressor()
49  grid_search = GridSearchCV(estimator=rf,
    ↪  param_grid=param_grid, cv=5)
50  grid_search.fit(X_train, y_train)
51
52  # Train ElasticNet
53  en = ElasticNetCV()
54  en.fit(X_train, y_train)
55
```

```python
56  # Get predictions
57  rf_pred = grid_search.predict(X_test)
58  en_pred = en.predict(X_test)
59
60  # Calculate squared residuals
61  rf_residual = (rf_pred - y_test) ** 2
62  en_residual = (en_pred - y_test) ** 2
63
64  # Perform Paired t-test
65  t_test = ttest_rel(rf_residual, en_residual)
66
67  ## Table 1: "Comparison of squared residuals from Random
    ↪   Forest and Elastic Net models"
68  df1 = pd.DataFrame({'RandomForest_mean_residual':
    ↪   [rf_residual.mean()],
69                      'RandomForest_std_residual':
                        ↪   [rf_residual.std()],
70                      'ElasticNet_mean_residual':
                        ↪   [en_residual.mean()],
71                      'ElasticNet_std_residual':
                        ↪   [en_residual.std()],
72                      't-statistic': [t_test.statistic],
73                      'p-value': [t_test.pvalue]},
                        ↪   index=["Result"]).round(2)
74  df1.to_pickle('table_1.pkl')
75
76  # Perform Wilcoxon signed-rank test
77  wilcoxon_test = wilcoxon(rf_residual - en_residual)
78
79  ## Table 2: "Wilcoxon signed-rank test on squared residuals
    ↪   from Random Forest and Elastic Net models"
80  df2 = pd.DataFrame({'z-statistic': [wilcoxon_test.statistic],
    ↪   'p-value': [wilcoxon_test.pvalue]},
    ↪   index=["Result"]).round(2)
81  df2.to_pickle('table_2.pkl')
82
83  # SAVE ADDITIONAL RESULTS
84  additional_results = {
85      'Total number of observations': len(data),
86      'Random Forest Score': grid_search.score(X_test, y_test),
```

```
87        'Random Forest Best Parameters': grid_search.best_params_,
88        'Elastic Net Score': en.score(X_test, y_test)
89  }
90  with open('additional_results.pkl', 'wb') as f:
91      pickle.dump(additional_results, f)
92
```

C.2   Code Description

The provided code performs data analysis on a dataset of pediatric patients
who received post-operative mechanical ventilation after undergoing surgery.
The goal is to determine the optimal tracheal tube depth (OTTD) for these
patients.

First, the code loads the dataset, which contains patient information
such as sex, age, height, weight, and the OTTD determined by chest X-ray.

Next, the code performs dataset preparations, including generating de-
scriptive statistics of height and age stratified by sex. These statistics are
saved in a pickle file called "table_0.pkl".

The code then preprocesses the dataset by creating dummy variables
for the categorical variable 'sex' and standardizing the values of continuous
features.

For the analysis, the dataset is split into training and testing sets. Two
models, Random Forest and Elastic Net, are trained on the training set.
Hyperparameter tuning is performed on the Random Forest model using
GridSearchCV to find the best combination of hyperparameters.

Predictions are then obtained using both the Random Forest and Elastic
Net models on the testing set. The squared residuals between the predicted
and actual OTTD values are calculated for both models.

A paired t-test is performed to compare the squared residuals from the
two models. The results, including means, standard deviations, t-statistic,
and p-value, are saved in a pickle file called "table_1.pkl".

Additionally, a Wilcoxon signed-rank test is conducted on the squared
residuals to evaluate if there is a significant difference between the two mod-
els. The results, including the z-statistic and p-value, are saved in a pickle
file called "table_2.pkl".

Finally, additional results are saved in a pickle file called "additional_results.pkl".
These results include the total number of observations, the Random Forest
score on the testing set, the best parameters found for the Random Forest
model, and the Elastic Net score on the testing set.

The code provides a comprehensive analysis of the dataset, comparing

11

the performance of the Random Forest and Elastic Net models in determining the optimal tracheal tube depth for pediatric patients.

C.3   Code Output

table_0.pkl

```
            ht        age_c
         mean   std  mean   std
sex
0-Female 65.40  18.7  0.73  1.40
1-Male   66.51  19.4  0.78  1.47
```

table_1.pkl

```
      RandomForest_mean_residual  RandomForest_std_residual
   ElasticNet_mean_residual  ElasticNet_std_residual  t-statistic  p-value
Result                        1.39                     3.57
    1.13                      2.3          2.16  0.03162
```

table_2.pkl

```
        z-statistic p-value
Result      19069.0  0.1302
```

additional_results.pkl

```
{
    'Total number of observations': 969,
    'Random Forest Score': 0.5635            ,
    'Random Forest Best Parameters': {'max_depth': 10, 'min_samples_split': 5,
    'n_estimators': 200},
    'Elastic Net Score': 0.6438            ,
}
```

D   LaTeX Table Design

D.1   Code

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

```python
1
2   # IMPORT
3   import pandas as pd
4   from my_utils import to_latex_with_note, format_p_value,
    ↪  is_str_in_df, split_mapping
5
6   # PREPARATION FOR ALL TABLES
7   shared_mapping = {
8    'ht': ('Height', 'Measured in centimeters'),
9    'age_c': ('Age', 'Measured in years'),
10   'sex': ('Sex', '0:Female, 1:Male'),
11   'std': ('Std. Dev.', None),
12   'mean': ('Mean', None)
13  }
14
15  # TABLE 0:
16  df = pd.read_pickle('table_0.pkl')
17  df.columns.names = ['Params', 'Stats']
18  df.index.names = ['Gender']
19
20  mapping = {k: v for k, v in shared_mapping.items() if
    ↪  is_str_in_df(df, k)}
21  mapping |= {
22   '0-Female': ('Female', None),
23   '1-Male': ('Male', None)
24  }
25  abbrs_to_names, legend = split_mapping(mapping)
26  df = df.rename(columns=abbrs_to_names, index=abbrs_to_names)
27
28  # Save as latex:
29  to_latex_with_note(
30  df, 'table_0.tex',
31  caption="Descriptive stats of Height and Age separated by
    ↪  Sex",
32  label='table:desc_stat_sex',
33  note="Table represents mean and std. dev. of Age and Height
    ↪  for both sexes.",
34  legend=legend)
35
36  # TABLE 1:
```

```python
37  df = pd.read_pickle('table_1.pkl')
38
39  # FORMAT VALUES
40  df['p-value'] = df['p-value'].apply(format_p_value)
41  df = df.T
42
43  # RENAME ROWS AND COLUMNS
44  mapping = {k: v for k, v in shared_mapping.items() if
    ↪  is_str_in_df(df, k)}
45  mapping |= {
46  'RandomForest_mean_residual': ('RF Mean Res.', None),
47  'RandomForest_std_residual': ('RF Std. Dev. Res.', None),
48  'ElasticNet_mean_residual': ('EN Mean Res.', None),
49  'ElasticNet_std_residual': ('EN Std. Dev. Res.', None),
50  't-statistic': ('T-stat', None),
51  'p-value': ('P-value', None)
52  }
53  abbrs_to_names, legend = split_mapping(mapping)
54  df = df.rename(columns=abbrs_to_names, index=abbrs_to_names)
55
56  # Save as latex:
57  to_latex_with_note(
58  df, 'table_1.tex',
59  caption="Comparison of Squared Residuals from RF and EN
    ↪  Models",
60  label='table:comp_residuals',
61  note="Table represents the comparison of Squared Residuals
    ↪  from RF and EN Models, including T-Statistic and P-value",
62  legend=legend)
63
64  # TABLE 2:
65  df = pd.read_pickle('table_2.pkl')
66  df.columns.names = ['Stats']
67  df.index.names = ['Result']
68
69  # FORMAT VALUES
70  df['p-value'] = df['p-value'].apply(format_p_value)
71
72  # RENAME ROWS AND COLUMNS
```

14

```
73  mapping = {k: v for k, v in shared_mapping.items() if
    ↪   is_str_in_df(df, k)}
74  mapping |= {
75  'z-statistic': ('Z-stat', None),
76  'p-value': ('P-value', None),
77  }
78  abbrs_to_names, legend = split_mapping(mapping)
79  df = df.rename(columns=abbrs_to_names, index=abbrs_to_names)
80
81  # Save as latex:
82  to_latex_with_note(
83  df, 'table_2.tex',
84  caption="Wilcoxon Signed-rank Test on Squared Residuals from
    ↪   RF and EN Models",
85  label='table:wilcoxon_test',
86  note="Table represents the result of Wilcoxon Signed-rank Test
    ↪   on Squared Residuals from RF and EN Models",
87  legend=legend)
88
```

D.2   Provided Code

The code above is using the following provided functions:

```
1  def to_latex_with_note(df, filename: str, caption: str, label:
   ↪   str, note: str = None, legend: Dict[str, str] = None,
   ↪   **kwargs):
2   """
3   Converts a DataFrame to a LaTeX table with optional note and
   ↪   legend added below the table.
4
5   Parameters:
6   - df, filename, caption, label: as in `df.to_latex`.
7   - note (optional): Additional note below the table.
8   - legend (optional): Dictionary mapping abbreviations to full
   ↪   names.
9   - **kwargs: Additional arguments for `df.to_latex`.
10
11  Returns:
12  - None: Outputs LaTeX file.
```

15

```
13   """
14
15   def format_p_value(x):
16     returns "{:.3g}".format(x) if x >= 1e-06 else "<1e-06"
17
18   def is_str_in_df(df: pd.DataFrame, s: str):
19     return any(s in level for level in getattr(df.index,
       ↪  'levels', [df.index]) + getattr(df.columns, 'levels',
       ↪  [df.columns]))
20
21   AbbrToNameDef = Dict[Any, Tuple[Optional[str], Optional[str]]]
22
23   def split_mapping(abbrs_to_names_and_definitions:
       ↪  AbbrToNameDef):
24     abbrs_to_names = {abbr: name for abbr, (name, definition) in
       ↪  abbrs_to_names_and_definitions.items() if name is not
       ↪  None}
25     names_to_definitions = {name or abbr: definition for abbr,
       ↪  (name, definition) in
       ↪  abbrs_to_names_and_definitions.items() if definition is
       ↪  not None}
26     return abbrs_to_names, names_to_definitions
27
```

D.3   Code Output

table_0.tex

```
\begin{table}[h]
\caption{Descriptive stats of Height and Age separated by Sex}
\label{table:desc_stat_sex}
\begin{threeparttable}
\renewcommand{\TPTminimum}{\linewidth}
\makebox[\linewidth]{%
\begin{tabular}{lrrrr}
\toprule
Params & \multicolumn{2}{r}{Height} & \multicolumn{2}{r}{Age} \\
Stats & Mean & Std. Dev. & Mean & Std. Dev. \\
Gender &  &  &  & \\
\midrule
```

```latex
\textbf{Female} & 65.4 & 18.7 & 0.73 & 1.4 \\
\textbf{Male} & 66.5 & 19.4 & 0.78 & 1.47 \\
\bottomrule
\end{tabular}}
\begin{tablenotes}
\footnotesize
\item Table represents mean and std. dev. of Age and Height for both sexes.
\item \textbf{Height}: Measured in centimeters
\item \textbf{Age}: Measured in years
\end{tablenotes}
\end{threeparttable}
\end{table}
```

table_1.tex

```latex
\begin{table}[h]
\caption{Comparison of Squared Residuals from RF and EN Models}
\label{table:comp_residuals}
\begin{threeparttable}
\renewcommand{\TPTminimum}{\linewidth}
\makebox[\linewidth]{%
\begin{tabular}{ll}
\toprule
 & Result \\
\midrule
\textbf{RF Mean Res.} & 1.39 \\
\textbf{RF Std. Dev. Res.} & 3.57 \\
\textbf{EN Mean Res.} & 1.13 \\
\textbf{EN Std. Dev. Res.} & 2.3 \\
\textbf{T-stat} & 2.16 \\
\textbf{P-value} & 0.0316 \\
\bottomrule
\end{tabular}}
\begin{tablenotes}
\footnotesize
\item Table represents the comparison of Squared Residuals from RF and EN
    Models, including T-Statistic and P-value
\end{tablenotes}
\end{threeparttable}
```

```latex
\end{table}


table_2.tex

\begin{table}[h]
\caption{Wilcoxon Signed-rank Test on Squared Residuals from RF and EN Models}
\label{table:wilcoxon_test}
\begin{threeparttable}
\renewcommand{\TPTminimum}{\linewidth}
\makebox[\linewidth]{%
\begin{tabular}{lrl}
\toprule
Stats & Z-stat & P-value \\
Result &   &  \\
\midrule
\textbf{Result} & 19069 & 0.13 \\
\bottomrule
\end{tabular}}
\begin{tablenotes}
\footnotesize
\item Table represents the result of Wilcoxon Signed-rank Test on Squared
    Residuals from RF and EN Models
\end{tablenotes}
\end{threeparttable}
\end{table}
```