Predicting Exacerbation and Associated Triage in COPD Patients


**Industrial representatives:**
Sumanth Swaminathan, Klajdi Qirko

Revon Systems Inc.

MPI 2017

June 19-23, 2017
Outline

1 Introduction

2 Supervised Machine Learning
   - Random Forests and Logistic Regression
   - Artificial Neural Network

3 COPD Exacerbation Predicting

4 Improvement and Future Work
Introduction

- Chronic Obstructive Pulmonary Disease (COPD)
  - eg. bronchitis, emphysema and pneumonia
- Application to give triage recommendation using baselines, symptoms and vitals
Introduction

LITERATURE & PANEL REVIEW

BASELINE

SYMPTOMS

VITAL SIGNS

SIMULATED PATIENT CASES

PHYSICIAN TRIAGE DATA

Triage Actions:
1) No Additional Medical Attention Required
2) Continue Normal Treatment & Return in 1-2 Days
3) Call Your Physician
4) Go to the Emergency Room

1) Likelihood of Exacerbation
2) Symptom Severity
3) Vital Sign Severity
4) Baseline Health Severity
Introduction

Frequency of exacerbation based on Symptom Severity

- Severity levels are assigned by doctors regarding 2300 patients.
- Frequency of an exacerbation is perfectly correlated with symptoms that a patient exhibits.
- If a patient has a symptom severity of 1 and 2, there is low chance that he/she has exacerbation, while
- If their severity is a 3-5 there is a large chance that they are experiencing exacerbation.
- Note that when the severity of the symptoms is 5 there is an almost zero chance of no exacerbation occurring.
How the model works

1. Determine vital signs, symptoms, and patient profile variables relevant to triage (from literature and physician panel review).

2. Computationally generate statistically & clinically relevant patient cases spanning the disease factors from step 1.

3. Collect & analyze triage/exacerbation data from physicians. Data is used for algorithm training & physician behavior studies.

4. Train various machine learning algorithms on physician data to optimize for accuracy, sensitivity, & specificity.

5. Validate machine learning prediction on the consensus decision of a panel of physicians triaging 100 patient cases.

6. Deploy prediction behind mobile application so that patients can enter data and receive automated recommendations.
Random Forests and Logistic Regression

- R & Python was used to build random forest models and logistic regression models.
- Missing values for continuous variables in the training and test data were handled by categorizing them.
- The random forest model compared using final triage I as target versus a weighted sum of final triage I and final triage II.
- The logistic regression model compared the target being final triage I versus weighting final triage I by the doctor’s confidence.
- Following are the comparison for accuracy and the confusion matrix.
Random Forests

- multiple decision trees
- combined to give *average* weight
Random Forests - results

First Triage Choice
Accuracy: 0.6337

<table>
<thead>
<tr>
<th>Reference</th>
<th>Prediction</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1 2 3 4</td>
</tr>
<tr>
<td>1</td>
<td>4 1 0 0</td>
</tr>
<tr>
<td>2</td>
<td>0 3 3 0</td>
</tr>
<tr>
<td>3</td>
<td>0 14 39 10</td>
</tr>
<tr>
<td>4</td>
<td>0 1 8 18</td>
</tr>
</tbody>
</table>

Weighted Triage Choices
Accuracy: 0.4455

<table>
<thead>
<tr>
<th>Reference</th>
<th>Prediction</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1 2 3 4</td>
</tr>
<tr>
<td>1</td>
<td>4 5 5 0</td>
</tr>
<tr>
<td>2</td>
<td>0 9 17 1</td>
</tr>
<tr>
<td>3</td>
<td>0 5 12 7</td>
</tr>
<tr>
<td>4</td>
<td>0 0 16 20</td>
</tr>
</tbody>
</table>
Figure 1: Feature importance for random tree, unweighted, using Python
Logistic Regression

- Fit a generalized linear model via penalized maximum likelihood using a multinomial distribution (glmnet package in R).
- Probabilities for possible outcomes of a single trial uses:
  \[
  \frac{e^{\beta_0 k + \beta_k^T x}}{\sum_{\ell=1}^K e^{\beta_0 \ell + \beta_\ell^T x}}
  \]
  
- Final triage I choice is weighted by doctor’s confidence.
## Logistic Regression - results

**First Triage Choice**  
**Accuracy:** 0.6139

<table>
<thead>
<tr>
<th>Reference</th>
<th>Prediction</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>8</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Weighted By Confidence**  
**Accuracy:** 0.6337

<table>
<thead>
<tr>
<th>Reference</th>
<th>Prediction</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>8</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>7</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>9</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Artificial Neural Network

Figure 2: Artificial Neural Network

(A) dendrites

B) cell body

terminal axon

C) synapse

D) synapses
MATLAB code was constructed to build a Neural Network having 3 hidden layers 20 neurons each.

It was trained keeping final triage I as the target and by substituting missing data as 0 and average of the variable.

The target was then setup as weighted average of Final Triage I and Final Triage II with confidence of doctors for both.

Following are the comparison tables for the four networks:
Artificial Neural Network

Table 1: Performance

<table>
<thead>
<tr>
<th></th>
<th>Final Triage I</th>
<th>Weighted Avg.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Missing data = 0</td>
<td>0.0772</td>
<td>0.0737</td>
</tr>
<tr>
<td>Missing data = Avg. of variable</td>
<td><strong>0.0631</strong></td>
<td><strong>0.0737</strong></td>
</tr>
</tbody>
</table>

Table 2: R value for training

<table>
<thead>
<tr>
<th></th>
<th>Final Triage I</th>
<th>Weighted Avg.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Missing data = 0</td>
<td>0.95698</td>
<td>0.95859</td>
</tr>
<tr>
<td>Missing data = Avg. of variable</td>
<td><strong>0.96391</strong></td>
<td><strong>0.95993</strong></td>
</tr>
</tbody>
</table>
Table 3: R value for testing

<table>
<thead>
<tr>
<th>Missing data</th>
<th>Final Triage I</th>
<th>Weighted Avg.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Missing data = 0</td>
<td>0.37522</td>
<td>0.29726</td>
</tr>
<tr>
<td>Missing data = Avg. of variable</td>
<td><strong>0.49822</strong></td>
<td>0.31508</td>
</tr>
</tbody>
</table>
Figure 3: Regression with missing data = Avg. of variable and final triage I
Figure 4: Performance of Training and Testing data

Best Training Performance is 0.063087 at epoch 64

Mean Squared Error (mse)

64 Epochs
Artificial Neural Networks Conclusions

- The best way of dealing with data is to substitute average of variable as missing data and setting the physicians first recommendation as the target.
- constructed Artificial Neural Network did not give efficient performance.
- Following approaches can be implemented to improve results:
  (a) normalizing the data
  (b) changing the activation function
Unsupervised Machine Learning

- trying to “understand” the data
- looking for structure or unusual patterns
- not looking for something specific (supervised)
- does not require labeled data
- evaluation usually indirect or qualitative
GOAL

- Cluster the patients based on app data without using Doctor recommendations using any unsupervised machine learning algorithm

Methods

- Choose a set of $k$ labels generated by REVON’s algorithm to attempt to match (Exacerbation $k = 2$ [0][1], Triage $k = 4$ [1][2][3][4], Triage $k = 3$ [1,2][3][4], etc.)
- Use $k$-means with the appropriate $k$ to generate our own clusters
- Compare our clusters with Revon clusters
Figure 5: Raw data
Figure 6: Clustered data
K-means

1. Fix an integer $k \geq 2$
2. Assign locations for the $k$ “means”
3. Assign each data point to a cluster based on the mean its closest to
4. Calculate the means as centroids of the clusters
5. Return to step 3 until convergence
Why use K-means?

Strength:
- Simple: easy to understand and to implement
- Computationally efficient $\approx O(Kn)$

Weakness:
- The user needs to specify $K$
- Highly sensitive to the choice of metric
- Sensitive to initial centroid locations
Results

- Output from comparison of four Triage groups with \( k \)-means with \( k = 4 \) clusters.
- Row index 1-4 is REVON’s algo triage category
- Column index is our “relabeled” \( k \)-means clusters
  - Since the label names from \( k \)-means are arbitrary, we check all permutations in order to maximize the overlap with REVON’s clusters (ie. the sum of the diagonal entries)
- Record the proportion of patients that were “correctly” classified (ie. matched REVON’s Triage predictions. Recall REVON’s algorithm matched doctor consensus more than any individual doctor.)

<table>
<thead>
<tr>
<th>cluster_pred</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>algs_pred</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>25</td>
<td>31</td>
<td>36</td>
<td>58</td>
</tr>
<tr>
<td>2</td>
<td>140</td>
<td>149</td>
<td>289</td>
<td>280</td>
</tr>
<tr>
<td>3</td>
<td>318</td>
<td>347</td>
<td>592</td>
<td>636</td>
</tr>
<tr>
<td>4</td>
<td>100</td>
<td>131</td>
<td>223</td>
<td>257</td>
</tr>
</tbody>
</table>
## Results

### Clusters assigned by Triage category

<table>
<thead>
<tr>
<th>Percent Overlap with our clusters</th>
<th>Raw Data</th>
<th>0-1 scale</th>
<th>z-score</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>k = 4 clusters</strong></td>
<td>[[1]], [[2]], [[3]], [[4]]</td>
<td>29.9</td>
<td>31.0</td>
</tr>
<tr>
<td></td>
<td>[[1]], [[2]], [[3]], [[4]]</td>
<td>33.8</td>
<td><strong>44.5</strong></td>
</tr>
<tr>
<td></td>
<td>[[1]], [[2]], [[3]], [[4]]</td>
<td>34.5</td>
<td>39.6</td>
</tr>
<tr>
<td></td>
<td>[[1]], [[2]], [[3]], [[4]]</td>
<td>34.7</td>
<td>44.5</td>
</tr>
<tr>
<td><strong>k = 3 clusters</strong></td>
<td>[[1]], [[2]], [[3]], [[4]]</td>
<td>52.2</td>
<td>68.0</td>
</tr>
<tr>
<td></td>
<td>[[1]], [[2]], [[3]], [[4]]</td>
<td>50.8</td>
<td><strong>73.0</strong></td>
</tr>
<tr>
<td></td>
<td>[[1]], [[2]], [[3]], [[4]]</td>
<td>51.7</td>
<td>60.9</td>
</tr>
</tbody>
</table>
Future Work/Challenges

- Weight the features differently according to their 'importance'
  - By using results from Supervised Group’s ‘feature importance’
  - By experimenting
- Try different metrics for $k$-means (we used Euclidean Distance)
- More data! There was a lot of missing data which forced us to bin and separate continuous random variable into binary variables
- Analyze our clusters to determine which features are most important for separation.
- Try other unsupervised methods
Predictive Model:

state variable $\mathbf{x} = (x_1, x_2, x_3)^T$
observation variable $\mathbf{z} = (z_1, z_2, z_3, z_4, z_5, z_6)^T$

vital signs:

\[
\begin{align*}
  z_1 & \leftarrow \text{temperature} \\
  z_2 & \leftarrow \text{pulse} \\
  z_3 & \leftarrow O_2 \text{ level}
\end{align*}
\]

symptoms:

\[
\begin{align*}
  z_4 & \leftarrow \text{severe cough} \\
  z_5 & \leftarrow \text{sputum} \\
  z_6 & \leftarrow \text{shortness of breath}
\end{align*}
\]

$z_i = H_i(\mathbf{x})$, where $H_i$ is the observation operator

for $i = \{1, 2, 3\}$ \quad $z_i = H_i(\mathbf{x}) = x_i + \xi_i$ \quad (1)

for $i = \{4, 5, 6\}$ \quad $z_i = H_i(\mathbf{x}) = h(g_j(\mathbf{x}) - t_i)$ \quad (2)

$\xi_i$ Gaussian noise, $h$ Heaviside function, $t_i = 0.5$ threshold
Predictive Model:

\[ \text{state variable } x = (x_1, x_2, x_3)^T \]
\[ \text{observation variable } z = (z_1, z_2, z_3, z_4, z_5, z_6)^T \]

vital signs: \[
\begin{align*}
&\{ z_1 \} \quad \leftarrow \text{temperature} \\
&\{ z_2 \} \quad \leftarrow \text{pulse} \\
&\{ z_3 \} \quad \leftarrow \text{O}_2 \text{ level}
\end{align*}
\]
symptoms: \[
\begin{align*}
&\{ z_4 \} \quad \leftarrow \text{severe cough} \\
&\{ z_5 \} \quad \leftarrow \text{sputum} \\
&\{ z_6 \} \quad \leftarrow \text{shortness of breath}
\end{align*}
\]

\[ z_i = H_i(x), \text{ where } H_i \text{ is the observation operator} \]

for \( i = \{1, 2, 3\} \quad z_i = H_i(x) = x_i + \xi_i \quad (3) \]

for \( i = \{4, 5, 6\} \quad z_i = H_i(x) = h(g_j(x) - t_i) \quad (4) \]

\( \xi_i \text{ Gaussian noise, } h \text{ Heaviside function, } t_i = 0.5 \text{ threshold} \)
Predictive Model:

state variable $\mathbf{x} = (x_1, x_2, x_3)^T$
observation variable $\mathbf{z} = (z_1, z_2, z_3, z_4, z_5, z_6)^T$

vital signs:

\[
\begin{align*}
\{z_1\} & \leftarrow \text{temperature} \\
z_2 & \leftarrow \text{pulse} \\
z_3 & \leftarrow O_2 \text{ level}
\end{align*}
\]

symptoms:

\[
\begin{align*}
\{z_4\} & \leftarrow \text{severe cough} \\
z_5 & \leftarrow \text{sputum} \\
z_6 & \leftarrow \text{shortness of breath}
\end{align*}
\]

$z_i = H_i(\mathbf{x})$, where $H_i$ is the observation operator

for $i = \{1, 2, 3\}$ \hspace{1cm} $z_i = H_i(\mathbf{x}) = x_i + \xi_i$ \hspace{1cm} (5)

for $i = \{4, 5, 6\}$ \hspace{1cm} $z_i = H_i(\mathbf{x}) = h(g_j(\mathbf{x}) - t_i)$ \hspace{1cm} (6)

$\xi_i$ Gaussian noise, $h$ Heaviside function, $t_i = 0.5$ threshold
COPD Exacerbation Predicting

Predictive Model:

state variable $\mathbf{x} = (x_1, x_2, x_3)^T$

observation variable $\mathbf{z} = (z_1, z_2, z_3, z_4, z_5, z_6)^T$

vital signs: symptoms:

\[
\begin{align*}
\{ z_1 \} & \leftarrow \text{temperature} \\
\{ z_2 \} & \leftarrow \text{pulse} \\
\{ z_3 \} & \leftarrow O_2 \text{ level}
\end{align*}
\]

\[
\begin{align*}
\{ z_4 \} & \leftarrow \text{severe cough} \\
\{ z_5 \} & \leftarrow \text{sputum} \\
\{ z_6 \} & \leftarrow \text{shortness of breath}
\end{align*}
\]

$z_i = H_i(\mathbf{x})$, where $H_i$ is the observation operator

for $i = \{1, 2, 3\}$ $z_i = H_i(\mathbf{x}) = x_i + \xi_i$ \hspace{1cm} (7)

for $i = \{4, 5, 6\}$ $z_i = H_i(\mathbf{x}) = h(g_j(\mathbf{x}) - t_i)$ \hspace{1cm} (8)

$\xi_i$ Gaussian noise, $h$ Heaviside function, $t_i = 0.5$ threshold
Modeling and Parameters

\[ g_j \text{ sigmoid function} \]

\[
g_1 = \left(1 + e^{-a_{11}x_1-a_{12}x_2-a_{13}x_3-a_{14}}\right)^{-1}
\]
\[
g_2 = \left(1 + e^{-a_{21}x_1-a_{22}x_2-a_{23}x_3-a_{24}}\right)^{-1}
\]
\[
g_3 = \left(1 + e^{-a_{31}x_1-a_{32}x_2-a_{33}x_3-a_{34}}\right)^{-1}
\]

We do logistic regression to find these 12 parameters, from real data. The time series data was organized from raw data and then interpolated to ensure that vital signs and symptoms occurred at the same time.
Kalman Filter

For each observation variable $z_i$ ($i = \{1, 2, \ldots, 6\}$), we apply a Kalman filter to the data time series:

Estimate the mean $\bar{x}$ and its covariance matrix $c$
\[
c = E[(x - \bar{x})(x - \bar{x})^T]
\]
for each observation variable $z_i$ ($i = \{1, 2, \ldots, 6\}$):

\[
\begin{align*}
\bar{x}|_{T+} &= \bar{x}|_{T-} + \kappa(\hat{z}_i|_T - H_i(\bar{x}|_{T-})) \\
c|_{T+} &= c|_{T-} - \kappa \mathcal{H} c|_{T-}
\end{align*}
\]

with $\mathcal{H} = \frac{\partial H}{\partial x}$ Jacobian, and $\kappa = c|_{T-} \mathcal{H}^T (\mathcal{H} c|_{T-} \mathcal{H}^T + \sigma^2 I)^{-1}$
An exacerbation is defined as exhibiting two symptoms for two consecutive days. We can calculate the probability by computing the following integral for two consecutive days:

\[
P[z_i] = \frac{1}{\pi^{3/2}|C|} \iiint_{\text{cond } B = \text{true}} e^{-x^T C^{-1} x} \, d^3x \tag{11}
\]

\[
\text{cond } B = (g_1 > 0.5 \land g_2 > 0.5) \lor (g_2 > 0.5 \land g_3 > 0.5) \\
\lor (g_1 > 0.5 \land g_3 > 0.5) \tag{12}
\]
Given the time series data for a particular user, we were only able to fit the sigmoid function parameters (4 of them) for the symptom coughing. The regression correctly predicted 12/18 of the symptoms.

The current implementation of the Kalman filter method performs at 37.5 percent accuracy.
Improvement and Future Work

- A future model may incorporate machine learning or Markov processes.
- More data per user (especially more symptom true values) at more regular intervals would be better for regression.
Thank you!
Questions?