Alexander Wan

500567571

GitHub Link: https://github.com/alexwan04/MRP-2018

Results

The final dataset used for this project had 16,371 records, 10 independent variables, and 1 response variable. In order to use tensor decomposition, the dataset was transformed so that every independent variable was a dimension and the values in the tensor were the 16,731 records of the response variable.

As mentioned in the exploratory data analysis, the tensor is a 2x12x7x2x59x2x2x2x2x2 sized object which results in the tensor having 634,368 available entries where 16,731 of the entries are non-zero. Since only 16,731 of the 634,368 sized-tensor was filled, this would be considered a sparse tensor.

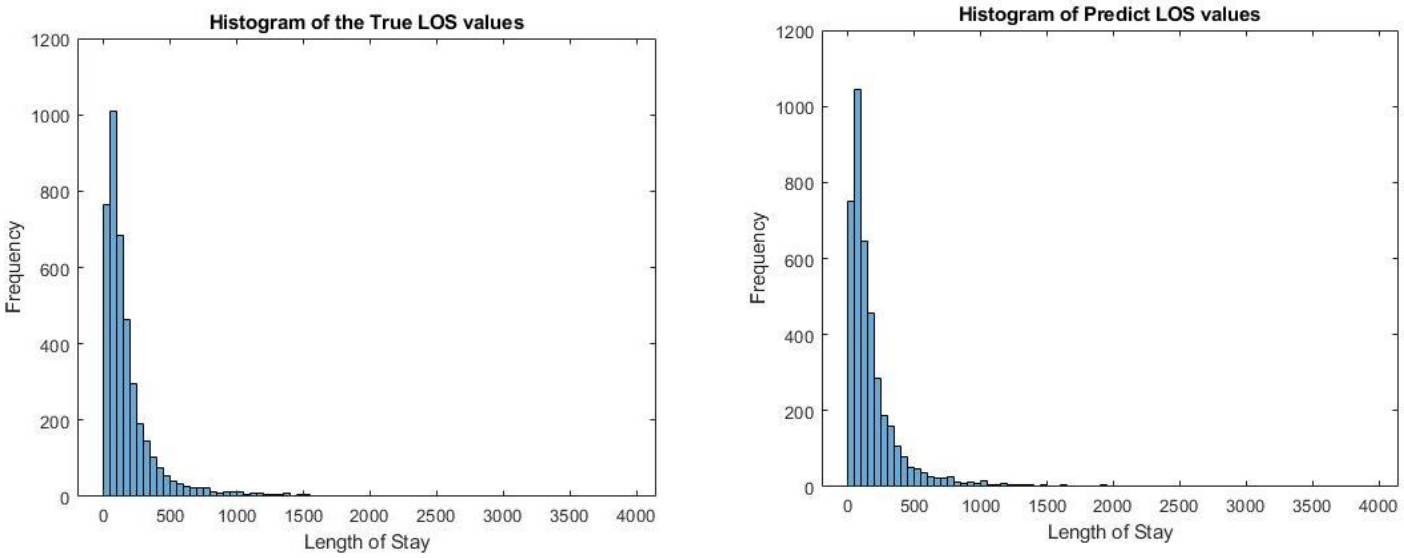
The type of tensor decomposition that was used for this experiment is known as Generalized Tensor Product (GTP) which is a form of Canonical Tensor Decomposition (CP). When creating training and testing sets in tensor decompositions, the method of splitting the data is slightly different than the normal way. Normally, after running the algorithm on the training set, one would have to run the algorithm on the testing set after. In tensor decomposition, one simply has to remove a portion of the dataset and after running the algorithm once, the new approximated tensor should already have attempted to predict the values that are in the testing set.

When running the model on the full dataset and then testing on the same dataset, the model was able to perfectly predict its values. This means the model is good enough to re-create the dataset after running it through the tensor decomposition. While it is good at predicting what values already in the dataset, it struggles in predicting variable combinations that did not appear in the training set.

In this experiment, the approximated tensor had an average error rate of 81.46 hours when using a 75%/25% training/testing set. This means that given a set of attributes of a patient's data, the algorithm can predict how long the patient will stay in the hospital with an accuracy of 81.46 hours. Given that the average length of stay in the dataset is 200.3 hours, this result is okay. Ideally, it would be better if the model could be more accurate, but there are some problems in the model which will be discussed later. This is interesting because when the model uses the same training and testing set, it is 100% accurate in its prediction. It is able to fully learn the length of stay patterns of known patients, but it struggles when it encounters new patients.

The figures below are the histograms of the length of stay from the test set and the predictions.

Figure 1: histogram of the length of stay from the test data (left) histogram of the length of stay from the predicted data (right).



Clearly, the graphs have a lot of similarity in that they are both skewed to the left with the second column having the highest frequency. They are generally similar but there are enough differences between the two that would cause a noticeable error. A problem is that when the length of stay values are low, the model predicts the length of stay to be 0 hours, which cannot happen.

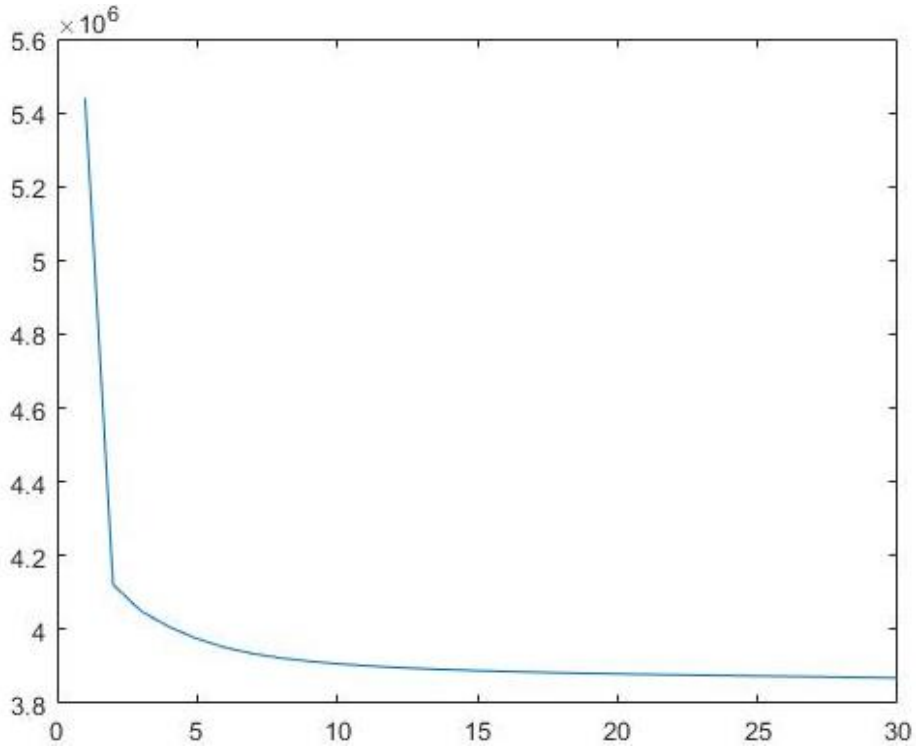


Figure 2: Plot showing that the model converges after around five iterations. At around ten iterations, the model has completely converged at around 3.9×10^6.

Five-Fold Cross-Validation

Training Set Name	Testing Set Name	Average Error in Hours
Train1	Test1	84.7356
Test1	Train1	82.5181
Train2	Test2	82.7492
Test2	Train2	84.5043
Train3	Test3	81.7357
Test3	Train3	85.5176
Train4	Test4	84.8428
Test4	Train4	80.4911
Train5	Test5	84.0643
Test5	Train5	83.2073

Table 1: The results of the five-fold cross-validation using the same error metric as the 75%/25% training/testing set test.

The average error for all the datasets in the five-fold cross-validation is 83.4366. This result is similar to the 75%/25% training/testing set test which legitimizes the results as it shows consistency.

Latent Dimensions

Since this dataset had 10 variables, there was 10 matching dimensions where each dimension's size was the range of the variable and 5. For example, the dimension that corresponds to the gender variable is 2x5 since there are two genders. The reason all the dimensions all have 5 for one of its sizes is because it represents the admitting service the patient used. There are 5 different admitting services levels: TMA, TMB, TMC, TMD, and TME. The admitting service dimension is also known as the topic for which each variable forms a matrix with. The model generates a latent tensor for each dimension where their values are like weights which can be analyzed to see which variables have the most effect on the approximated tensor.

The values of each latent dimension are posted in the appendix section which is a text file in GitHub. The gender variable has the most effect on the model as all of its values are larger than 1. The other 9 variable's latent factor values are mainly single digit values.

Gender

Topic\Dimension	Male	Female
TMA	456.6877	203.6788
TMB	277.7146	587.9269
TMC	1492	1361.6
TMD	366.8361	69.1930
TME	822.0103	632.1651

Table 2: The values of the latent tensor for the gender variable

The biggest values seem to be from topic 3: TMC. This would suggest that the activation levels of the model are highest when the topic is TMC.

One interesting note is that it seems that latent values for males seem to be greater than for females outside of TMB. This could mean that for the most part, males stay in the hospital longer than females.

The Rest of the Variables

The values of all the other variables are much smaller. For the time variables, month and day of week, this would suggest that the month or day of week of when a patient was admitted has very little effect on the model.

Additionally, the five variables that count body parts that have been tested (CT head, POR chest, RAD chest, RAD lower EXT, and RAD pelvis hip) are variables that the hospital specifically requested to be used in the model. Out of those five it seems that RAD pelvis hip count has the most effect as their values are mainly greater than one while all other values are less than one. There's a greater effect when a patient does not have any of these variables checked off which would make sense as majority of the records do not have these indicator variables checked off. The combination that gives the biggest value is for a patient does not have RAD Pelvis Hip checked off and has admitting service = TMA. Since RAD pelvis hip appears the least in the dataset (506 records have RAD pelvis hip checked-off compared to RAD chest which has 6998 records checked-off and has the highest count of the 5 variables) it would make sense that it has the greatest effect on the model when not checked compared to the other four variables. Another way of interpreting this is that the activation levels tend to be higher when a patient does not have RAD pelvis hip checked-off.

Comparison with Other Machine Learning Algorithms

In addition to tensor decomposition, basic linear regression and singular value decomposition was also tested on the dataset for comparison. Linear regression was chosen because it is the simple machine learning algorithm for basic testing. Singular value decomposition was chosen because it is like a lower dimensional version of tensor decomposition. The results are as follows:

Algorithm	Average Error
Tensor Decomposition	81.46 (83.44 for 5-fold CV)
Linear Regression	154.33
Singular Value Decomposition	203.46

Table 3: Average error for the three algorithms tested on the hospital data

Tensor decomposition has much better results than the other two algorithms. While the linear regression and SVD models were simple models, the result is that they do not do a good job in predicting patient's length of stay. They were both noticeably worse than tensor decomposition and considering that the average length of stay is 110 hours, the error is worse than the average. Since tensor decomposition is the most accurate model, it seems that tensor decomposition is most suitable despite the fact that it does not do a good job of predicting. One thing to note is that none of them have great results so it could be a dataset problem. It seems that all algorithms struggle with prediction on this dataset.

```
Call:
lm(formula = LOS ~ ., data = smh_train)

Residuals:
   Min     1Q Median     3Q    Max
-367.8 -132.5  -80.8   20.1 3724.9

Coefficients:
                    Estimate Std. Error t value Pr(>|t|)
(Intercept)          157.9534    12.7883  12.351  < 2e-16 ***
Gender2               11.3688     5.2286   2.174 0.029697 *
TriageMonth2         -28.6819    12.5946  -2.277 0.022784 *
TriageMonth3         -22.6896    12.6094  -1.799 0.071975 .
TriageMonth4          -9.7516    13.3261  -0.732 0.464322
TriageMonth5         -24.5373    12.3032  -1.994 0.046133 *
TriageMonth6         -18.4767    12.5740  -1.469 0.141741
TriageMonth7         -25.2335    12.2996  -2.052 0.040234 *
TriageMonth8          -8.9984    12.3542  -0.728 0.466402
TriageMonth9         -11.1658    12.3628  -0.903 0.366447
TriageMonth10          5.2766    12.4541   0.424 0.671806
TriageMonth11        -10.0092    12.6534  -0.791 0.428940
TriageMonth12        -27.3427    12.3735  -2.210 0.027139 *
TriageDayOfWeek2      19.0430     9.9574   1.912 0.055842 .
TriageDayOfWeek3      24.6234     9.8777   2.493 0.012686 *
TriageDayOfWeek4      23.7064     9.9974   2.371 0.017743 *
TriageDayOfWeek5      30.5689    10.1183   3.021 0.002523 **
TriageDayOfWeek6      38.5687    10.1535   3.799 0.000146 ***
TriageDayOfWeek7      17.9258    10.4862   1.709 0.087389 .
NON_GIM_ER.CONSULT2   16.8796     7.0629   2.390 0.016867 *
WaitTime.to.Admit     -1.3737     0.7359  -1.867 0.061959 .
CT_HEAD_Count2        50.4980     6.4233   7.862 4.11e-15 ***
POR_CHEST_Count2      57.2651     8.1937   6.989 2.91e-12 ***
RAD_CHEST_Count2      33.9010     5.5158   6.146 8.18e-10 ***
RAD_LOWER_EXT_Count2  53.2371    15.0607   3.535 0.000410 ***
RAD_PELVIS_HIP_Count2 87.2218    15.2969   5.702 1.21e-08 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 286.3 on 12252 degrees of freedom
Multiple R-squared:  0.0177,    Adjusted R-squared:  0.0157
F-statistic: 8.831 on 25 and 12252 DF,  p-value: < 2.2e-16
```

Figure 3: Summary from the Linear Regression model

With a p-value of 0.05, it seems that only the month and wait time to admit variables are considered not significant. For a more optimal model, there would need a change in the data with more data processing. For example, removing those two variables could improve the performance of the model.

Improvements for the Future

It seems that the results of the model were not good. As the model predicts a patient's length of stay with error of around 80 hours, it would need improvements before it can become reliable. One major improvement would be to have better hardware. The main reason that the model had 10 variables/ dimensions/ modes is that adding more will cause the computer to run out of memory. Having a better computer would allow the model to fit more variables/ dimensions/ modes. On the other hand, too much sparsity can also be an issue. As more variables/ dimensions/ modes gets added to the model, the tensor becomes larger and larger which would cause the tensor to be very sparse. If it is too sparse, it can be hard to predict as the prediction model may not have enough information in certain areas of the tensor to accurately predict a value in that area.

Since medical data tends to have a large number of variables, tensor decomposition may not be suited for it. The problem is that the ratio of variables to datapoints in the medical data is not good for tensor decomposition as the tensor will be too sparse for the model to accurate predict. Having more variables would be better from the hospital's point of view as it can look at how any of the variables can affect the model even if the variable is considered insignificant from a statistically point of view. A solution may be to use another machine learning algorithm such as neural networks which supports having a lot of variables.

Additionally, the data processing needs improvement. The files that contained the data were very messy and unorganized. They were unsuitable for regular data analysis as they contained a lot of text which would require sentiment analysis to be included into regular machine learning algorithms. They also had hard to understand values like ranges (ex. A cell could contain the value 5-8).