Homework 1 Answers

Kelly Scott Sims

Q2.b

Metric	Deceased patients	Alive patients	Function to complete
Event Count 1. Average Event Count	982.014	498.118	event_count_metrics
2. Max Event Count	8635	12627	
3. Min Event Count	1	1	
Encounter Count 1. Average Encounter Count	23.038	15.452	encounter_count_metrics
2. Max Encounter Count	203	391	
3. Min Encounter Count	1	1	
Record Length 1. Average Record Length	127.532	159.2	record_length_metrics
2. Max Record Length	1972	2914	
3. Min Record Length	0	0	

Table 2: Descriptive statistics for alive and dead patients

Q4.1b

Model	Accuracy	AUC	Precision	Recall	F-Score
Logistic Regression	0.954	0.945	0.9869	0.8988	0.9408
SVM	0.9940	0.9945	0.9882	0.9970	0.9925
Decision Tree	0.7763	0.7475	0.7921	0.6011	0.6835

Table 3: Model performance on training data

Q4.1c

Model	Accuracy	AUC	Precision	Recall	F-Score
Logistic Regression	0.7381	0.7375	0.6804	0.7333	0.7059
SVM	0.7381	0.7389	0.6768	0.7444	0.7089
Decision Tree	0.6714	0.6569	0.6329	0.5555	0.5917

Table 4: Model performance on test data

Q4.1d

Based on the performance metrics on training and test data, it is very evident that the models are experiencing **High Variance**. The models are overfitting the data. This is mostly due to the amount of features vs the amount of observations. When there are far greater features than observations, models will tend to fit to the randomness that just so happens to be in the training data. Because of this, they don't generalize very well to new unseen data, and the randomness that comes with it.

Therefore, one could aggregate more observation (data points). It is generally a good practice to have **observations > features**. If collecting more data is not feasible, then dimensionality can be reduced using PCA. By ranking the spread of variance in eigenvectors by their eigenvalues, the most important eigenvectors can be selected, consequently reducing the amount of features to combat the overfitting problem.

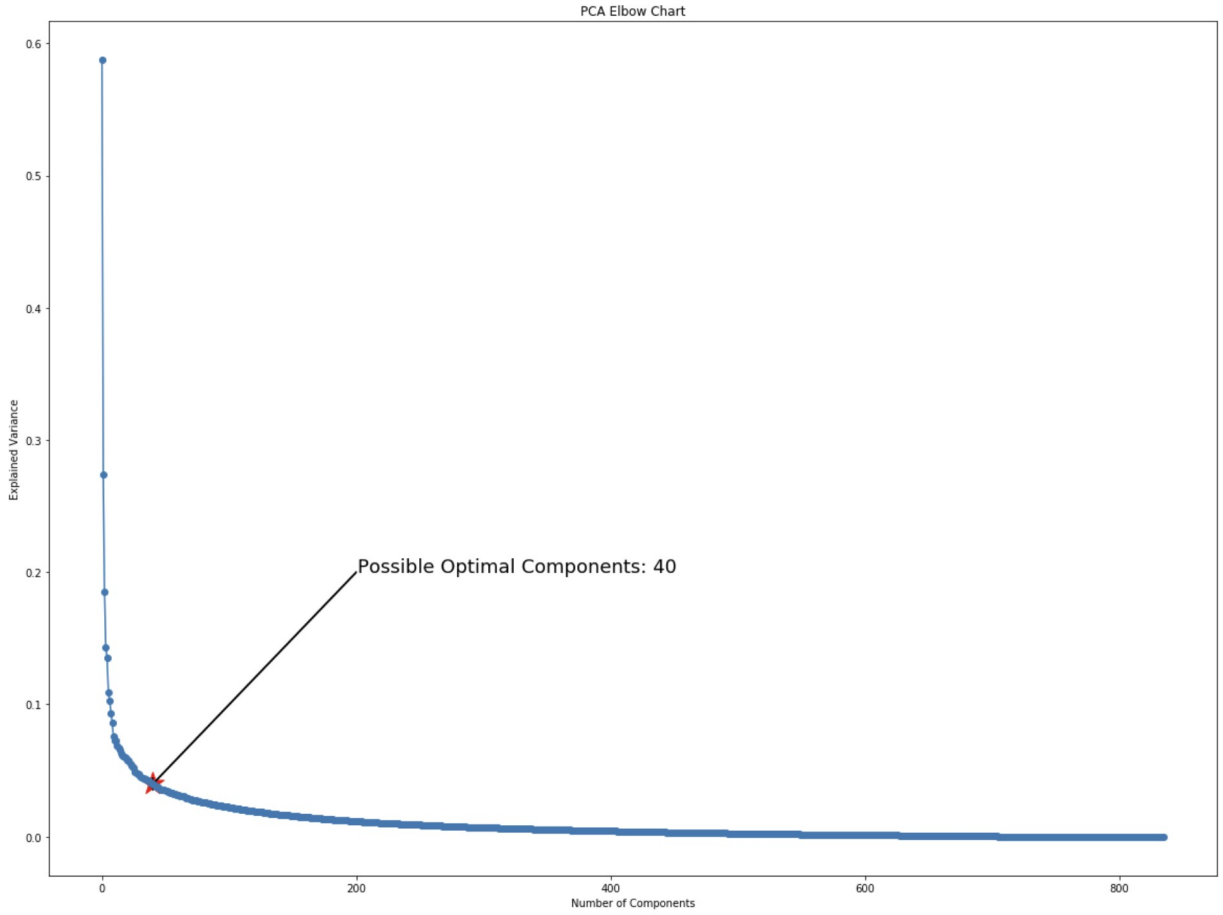
Finally, if model performance doesn't increase simply by reducing dimensionality, models can be tuned by adjusting hyper parameters. E.g. the regularization constant for a logistic regression model can be tuned to reduce the amount of high variance seen.

Q4.2b

CV strategy	Accuracy	AUC
K-Fold	0.7213	0.7075
Randomized	0.7357	0.7188

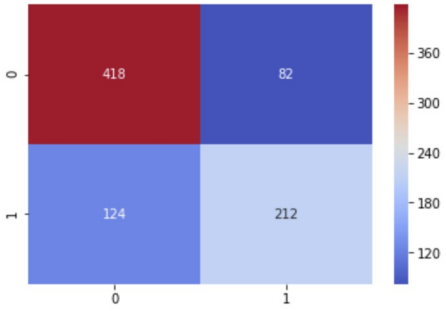
Q4.3b

The first approach in building my model was dimensionality reduction utilizing PCA. In order to do so, I first needed to figure out how many eigenvectors (features) to reduce to. This was done with a “PCA Elbow Chart”, where the elbow of the chart is usually deemed the most optimal amount of eigenvectors.



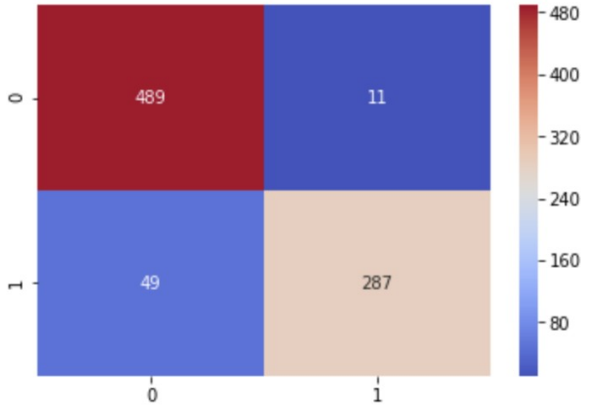
The chart showed about 40 eigenvectors to be optimal. However, since it was such a wide sweeping elbow, I employed Gridsearch CV while training my model. Gridsearch allows one to iteratively try out various hyper-parameter values together and returns the best combination of said parameters. Employing Gridsearch, a SVM model was built while testing out eigenvectors ranging from 20 to 40 and C parameter ranging 0.01 to 1000. The results were less than stellar.

```
Full Train Data Accuracy: 0.7535885167464115
AUC: 0.7334761904761905
```



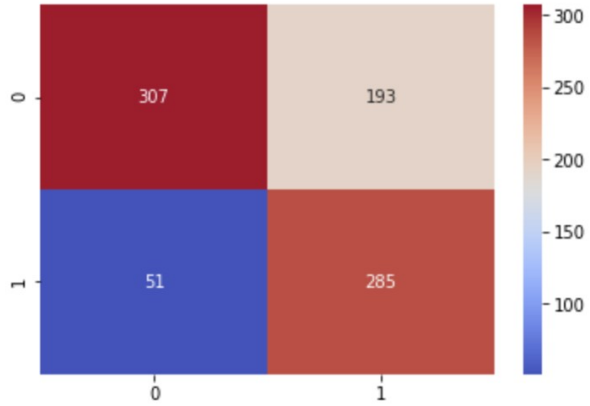
The next strategy was to just tune the best model using all features. Gridsearch again was employed to build: SVM, Random Forest, Logistic Regression, Gradient Boost, Naive Bayes, K-Nearest neighbors and Gaussian Process Classification model. A wide range of hyperparameters and optimization algorithms were employed in the Gridsearch CV to give the following results.

```
Full Train Data Accuracy: 0.9282296650717703
AUC: 0.9160833333333332
```



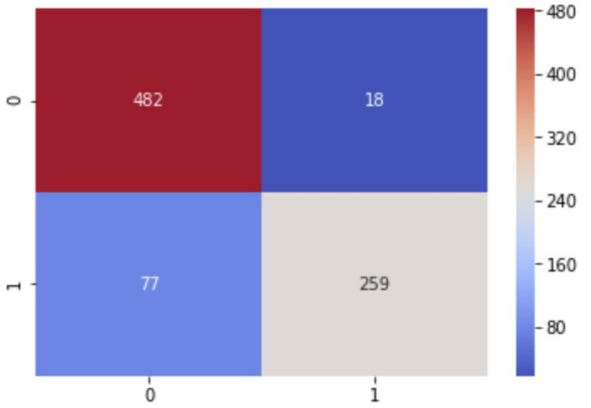
Naive Bayes

```
Full Train Data Accuracy: 0.7081339712918661
AUC: 0.7311071428571428
```



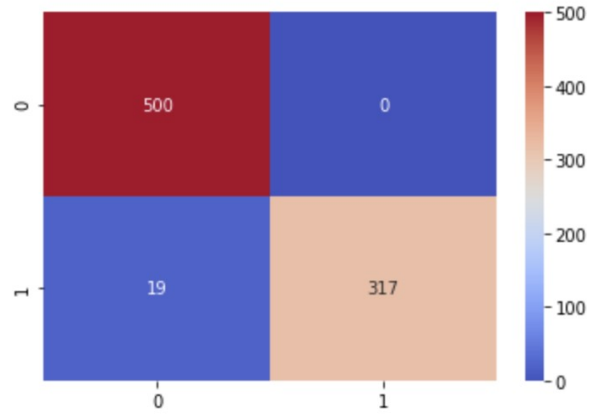
knn

```
Full Train Data Accuracy: 0.8863636363636364
AUC: 0.8674166666666667
```



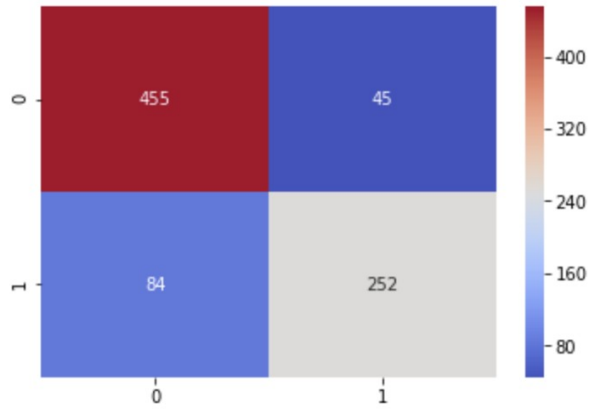
logistic regression

```
Full Train Data Accuracy: 0.9772727272727273
AUC: 0.9717261904761905
```



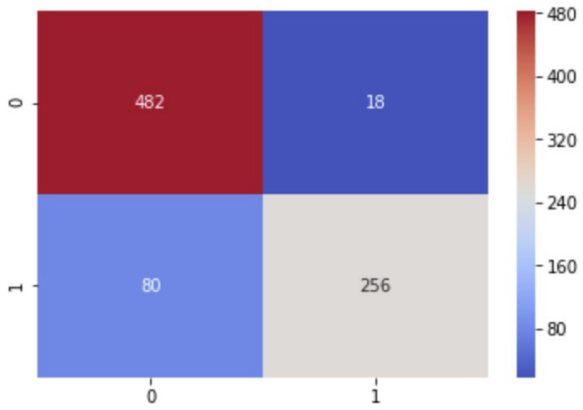
gradient boost

Full Train Data Accuracy: 0.8456937799043063
AUC: 0.8300000000000001



Random Forest

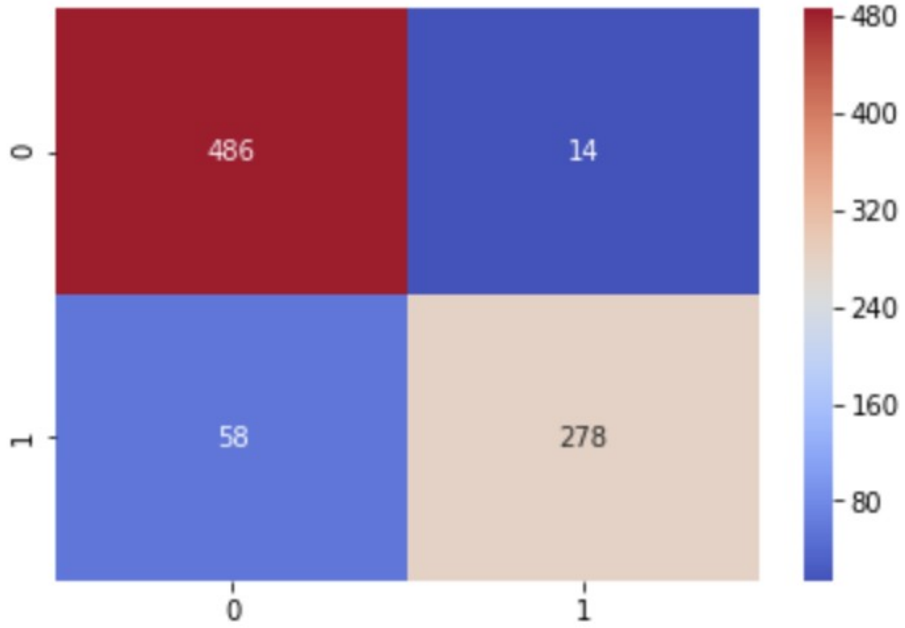
Full Train Data Accuracy: 0.8827751196172249
AUC: 0.8629523809523809



SVM

The results for all except the KNN model were respectable. There was still a fair amount of overfitting however. To combat that, I decided to use all the models in an Ensemble Voting model, where each of the aforementioned models will cast their vote on the prediction. Majority vote wins. The results of the ensemble model wasn't as great as the Gradient boosted model. But the high accuracy and AUC in the boosted model was clearly due to high variance.

Full Train Data Accuracy: 0.9138755980861244
AUC: 0.899690476190476



Final Ensemble Model (Voting Model)