Homework 2: PCA vs LDA

Created	@September 12, 2023 6:54 PM
Class	Predictive Modelling
Type	Homework
Author	Ⓐ Alejandro Diaz

💡 In this exercise, the dataset that will be used is “Heart_Disease”. This database contains 76 attributes, but all published experiments refer to using a subset of 14 of them. In particular, the Cleveland database is the only one that has been used by ML researchers to date. The "goal" field refers to the presence of heart disease in the patient. It is integer valued from 0 (no presence) to 4. Experiments with the Cleveland database have concentrated on simply attempting to distinguish presence (values 1,2,3,4) from absence (value 0). The dataset can be obtained in the repository “UC Irvine Machine Learning Repository” (https://archive.ics.uci.edu/dataset/45/heart+disease).

💡 For this exercise you can be use any of the files namely “processed.xxxxx.data”. You should consider that there are some differences between the classes in some of those files.

For this exercise, I am going to be using the dataset contained in `processed.cleveland.data`,

The subset of the variables that this processed data contains is the following:

> Attribute Information:
> -- Only 14 used
> -- 1. #3 (age)
> -- 2. #4 (sex)
> -- 3. #9 (cp)
> -- 4. #10 (trestbps)
> -- 5. #12 (chol)
> -- 6. #16 (fbs)
> -- 7. #19 (restecg)
> -- 8. #32 (thalach)
> -- 9. #38 (exang)
> -- 10. #40 (oldpeak)
> -- 11. #41 (slope)
> -- 12. #44 (ca)
> -- 13. #51 (thal)
> -- 14. #58 (num)

The goal in this exercise is to train different models of classification and compare them with metrics of: accuracy, precision and recall. The models to be compared are:

- Logistic Regression Model (no data transformation)
- Logistic Regression Model with PCA transformation
- Logistic Regression Model with LDA transformation
- LDA Model

Data Preparation

In this dataset, there are some missing values marked with a character “?”. At the moment of import, I am marking the “?” character as “na” values:

Column	Type	Missing values	Present values	Unique values	Min value	Max value
age	float64	0	303	41	29	77
sex	float64	0	303	2	0	1
cp	float64	0	303	4	1	4
trestbps	float64	0	303	50	94	200
chol	float64	0	303	152	126	564
fbs	float64	0	303	2	0	1
restecg	float64	0	303	3	0	2
thalach	float64	0	303	91	71	202
exang	float64	0	303	2	0	1
oldpeak	float64	0	303	40	0	6.2
slope	float64	0	303	3	1	3
ca	float64	4	299	4	0	3
thal	float64	2	301	3	3	7
num	int64	0	303	5	0	4

As seen, there are four missing values in “ca” and two missing values in “thal”, since they represent 1.9% of the data, we are just going to remove those rows.

In the context of these models, our target variable “num." This variable exhibits five potential outcomes, where 0 signifies the absence of heart disease, and values 1 through 4 represent distinct types of heart diseases. In this specific scenario, our primary interest lies in discerning the mere presence or absence of heart disease. Consequently, we have consolidated the categories corresponding to types 1 through 4 into a single category, labeled as "1," thus resulting in a binary classification task.

Train and Test subsets

💡 Using the data set mentioned, develop the following points.

💡 1. Create two subsets of data, where the first one will be used for the training process and the second one for the testing process. It is important that the same data subsets are used for all subsequent models.

The separation into subsets for training and testing is done with sklearn.model_selection.train_test_split(), with 30% of the data for testing, using a random seed of “0”.

Classification Models

Base Logistic Regression Model

> 💡 2. Train a logistic regression model to estimate the feature “target” using all the pixels in the image as input variables. Get the accuracy values or another metric (precision, recall, …) to evaluate the model's performance in training and testing.

For our base model, there are no transformations to the data before training the logistic regression model.

Results:

Metric	Training	Testing
Accuracy	0.87	0.84
Recall	0.80	0.76
Precision	0.90	0.89

The intercept for this model is: 0.21726

And the coefficients are:

0	-0.042445
1	0.600125
2	0.40486
3	0.027931
4	0.002266
5	-0.358169
6	0.096028
7	-0.045816
8	0.970768
9	0.343609
10	0.180966
11	0.935319
12	0.361089

PCA + Logistic Regression

> 💡 3. Considering the data set obtained in point 1, perform the dimension reduction using the PCA method. With the variables resulting from the reduction process, train a new logistic regression model and calculate the same metrics used in point 2.

When we do Principal Component Analysis on the predictors before passing it to the logistic regression, we get the following variance explained by each component, and the cumulative:

Component #	Explain Variance ratio	Cumulative
1	0.738612	0.738612
2	0.15263	0.891242
3	0.089557	0.980799
4	0.016954	0.997753
5	0.001065	0.998818
6	0.000336	0.999154
7	0.000258	0.999411

Component #	Explain Variance ratio	Cumulative
8	0.000217	0.999628
9	0.000182	0.99981
10	0.000066	0.999876
11	0.000048	0.999924
12	0.000041	0.999965
13	0.000035	1

Based on the cumulative variance, it seems to be that it's worth taking only the first 3 components, since they explain up to 98% of the variance. So we are training the Logistic Regression only using three components. (You can find PCA transformation matrix as a pickle file as pca_matrix.pickle)

These are the results of doing so:

Metric	Training	Testing
Accuracy	0.74	0.66
Recall	0.68	0.60
Precision	0.73	0.64

There is a significant reduction of the metrics when doing the testing. Even during training, recall score decrement quite significantly resulting in an increment of false negatives. This could be potentially dangerous, as we could end up cataloging a patient as "healthy" when there's potentially a heart issue.

Even when 98% of the variance, PCA transformation is behaving poorly, mainly because it is not taking the target variable at the moment of the fitting, so we are loosing important information in the predictors.

The resulting logistic regression model is:

	coefficients
0	0.004737
1	0.050546
2	0.017431

Intercept: - 0.173801

LDA Transformation + Logistic Regression

> 💡 4. Considering the data set used in point 1 again, perform a variable reduction by LDA transformation. With the variables resulting from the transformation process, train a new logistic regression which must be evaluated with the same metrics as the previous models.

Similarly as we did with PCA, we are now transforming the data with a Linear Discriminant Analysis to see its effects on the logistic regression.

LDA is only returning one component, since the other components contribution to the variance must be 0. This is the component found by LDA:

0	0.00776
1	1.074507
2	0.790593
3	0.037402
4	0.003123
5	-0.768503
6	0.143152

7	-0.023521
8	1.561801
9	0.358082
10	0.486663
11	0.903082
12	0.491127

If we use this to transform the data, that means we only keep one variable as input to train the Logistic Regression. Let's see the results of doing so:

Metric	Training	Testing
Accuracy	0.86	0.84
Recall	0.83	0.76
Precision	0.86	0.89

This is behaving very similar to the base line, but using only one variable for training. This is a decrement from 13 to 1 variable, indicating that LDA is taking into consideration only the relevant variance of all the variables into just one.

This is the model proposed:

Coefficient	1.778279
Intercept	0.142786

LDA Model

Considering the data set used in point 1 again; train a LDA model and evaluate the model with the same metrics

Now instead of using LDA just as a transformation, let's see how it performs doing actual predictions:

Metric	Training	Testing
Accuracy	0.86	0.82
Recall	0.81	0.71
Precision	0.88	0.88

If we see this, it's behaving just below LDA + Logistic Regression, but it is still a good linear model.

This is the model found by LDA:

	Coefficient
0	0.00776
1	1.074507
2	0.790593
3	0.037402
4	0.003123
5	-0.768503
6	0.143152
7	-0.023521
8	1.561801
9	0.358082
10	0.486663
11	0.903082

	Coefficient
12	0.491127

Intercept: - 10.995195

Results Discussion

This is a summary of the results:

	Base		PCA + LR		LDA + LR		LDA Model
	Train	**Test**	**Train**	**Test**	**Train**	**Test**	**Train**
Accuracy	0.87	0.84	0.74	0.66	0.86	**0.84**	0.86
Recall	0.80	0.76	0.68	0.60	0.83	**0.76**	0.81
Precision	0.90	0.89	0.73	0.64	0.86	**0.89**	0.88

As seen in the table, transforming with PCA showed to be the worst strategy, it gave lower scores for all accuracy, recall and precision scores in both train and test sets.

Using a combination of LDA and a Linear Regression is arguably the best option. It behaves just below base Linear Regression scores for training, and for testing it gave the same scores. The major advantage we have with LDA + LR is that it showed to give the same results with only one variable, instead of the base line which is using 13 to accomplish the same. This reduction of dimensions leads to a simpler model that can potentially provide better performance results. On the other hand, if we want to skip the logistic regression all together, and only train with LDA, the model itself proposed by LDA is giving quite good results, with a narrow difference as doing it with the mixed model. It is just left behind by 2% in accuracy, 5% in recall, and 1% in precision.