Probabilistic Graphical Models
Project1
Respiratory Rate Estimation

I. Brief description

The project aims at estimating the respiratory rate of an individual using the inertial, heart rate and temperature measurements. The inertial data comes from accelerometer and gyroscopic values from a wristband sensor while the heart rate and body temperature values are obtained from the chest strap sensor. For estimation we initially proposed 3 models: Stepwise regression, Ridge regression and Neural networks. Among these we finalized down on Ridge regression and Neural Networks based on the performance and calculation time.

II. Dimensionality Reduction: Issues and Selection Methodology[12]

The main idea of principal component analysis (PCA) is to reduce the dimensionality of a data set with many variables correlated with each other, while retaining the variation present in the dataset, up to the maximum extent. PCA does this by transforming the existing set of variables to a new set of variables (called principal components) which are orthogonal. The order of this principal component vectors is such that the retention of the variation in the original vectors decreases as we move down the order.

In our case we carried out dimensionality reduction in two stages: First we manually computed the correlation between all the features and eliminated highly correlated features; Second we passed this reduced number of features as an input to PCA for further reduction in dimension. For the former manual method of reduction,we found the correlation between each features using the toolbox numpy[13]. We considered values of correlation of 0.9 and above to be highly correlated. We found a set of 6 features to be highly correlated with each other and another set of 4 features with high correlation. So basically, instead of considering all these 10 features, we took only one feature from each set to predict the output reducing the number of features from 52 to 44. The purpose behind doing manual reduction using correlation was to eliminate the need to perform additional computations by PCA. Further, we used PCA to ensure optimum dimensionality reduction. For Ridge regression, PCA resulted into 42 features. To obtain this value, we varied the number of features after PCA reduction from 1 to 44 and observed the value of R-squared for the respective values which is as shown in Figure 1. It is evident that the R-squared value is maximum for features from 42 to 44. However, we chose the least number of features among these values i.e. 42. So the optimum number of features was selected as 42. For Neural Network we found the best possible PCA reduced feature by manual trial and error. The number of features found to give least possible error were 20.

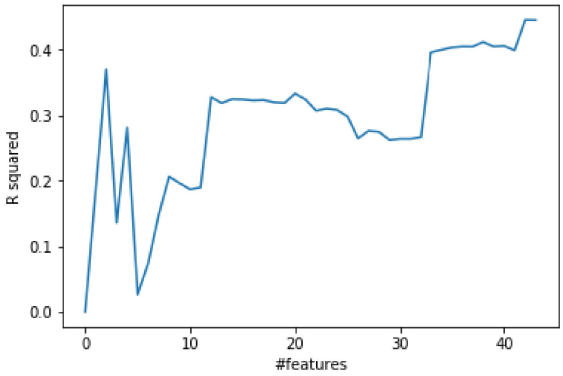


Fig. 1. Selection of PCA based on R2 metric for Ridge Regression

III. Description of Methods for Respiratory Rate Estimation

We developed methodologies for prediction of respiratory rate using two methods from class of supervised learning: Ridge Regression and Neural Networks.

A. Ridge Regression[5][11]

Ridge Regression is a form of linear regression, which is used in cases where independent variables are highly correlated i.e. the data suffers from multicollinearity. In a linear regression with least square solution the estimates are unbiased (the expected values of the parameters are the true values), and of all the unbiased estimators, it gives the least variance (most precise answer). However, when the two or more predictor variables are strongly correlated there is a large variance in the final parameter estimates. Ridge regression solves this problem by acting as an estimator which is no longer unbiased but has considerably less variance than least-square estimators. How is it applicable in our case? Currently with the features provided we are not sure about the degree of correlation between the predictors. Hence, we intend to find the correlation between the predictors with the available data and if it is observed that there exists some significant correlation between significant predictors, then ridge regression could act as a good estimator.

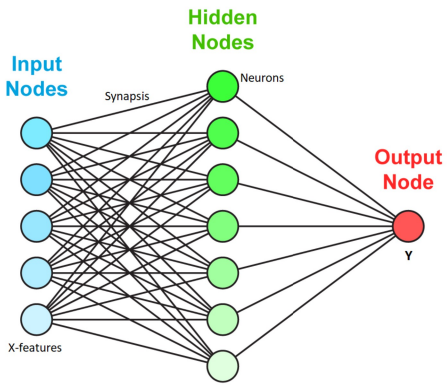


Fig. 2. Schematic: ANN

The dimensionality of the features can be reduced using PCA before applying ridge regression. Although manual correlation check and PCA eliminates those features which are highly correlated, ridge regression can be used since we have around 42 features left for the prediction which is still a significant number. This method chooses the β to penalize in such a way that features that contribute less to the output undergo more penalization. This reduces the complexity of the model without compromising on the correctness of the prediction.

$$\beta = argmin(y - X\beta)^2 + \lambda * \beta^2 \quad (1)$$

Where λ is the Hyper parameter, β is the model parameter, X is the input data and Y is the prediction.

B. Artificial Neural Networks(ANN)[6]

Neural Networks are non-linear statistical data modeling tools. A general schematic of ANN is as shown in Fig. 2. They are used to model complex relationships between inputs and outputs or to find patterns in data. Neural Networks can be considered as universal approximators which can model any complex non-linear function. Since the data we are dealing with, varies non linearly with the output(respiratory rate), Neural Networks can be considered as a potential candidate for model selection. To reduce the dimensions of the data, we applied PCA to the training data and reduced the number of features to 20. This reduces the number of computations the network has to perform.

1) Preprocessing: We standardized the feature by removing the mean and scaling the variance to unit value. Centering and scaling happen independently on each feature by computing the relevant statistics on the samples in the training set. Mean and standard deviation are then stored to be used on later data using the transform method. If a feature has a variance that is orders of magnitude larger than others, then it might dominate a node and make the network unable to learn the features correctly.

IV. Selection of Hyper Parameters

Hyper parameters are parameters whose values are kept fixed before the learning process and will not be changed during the whole process. Given these hyper parameters, the algorithm learns the value of parameters from the data.

A. Selection of k for Cross Validation

For testing, we used k-fold Cross validation method. The prediction values after each fold is concatenated and compared with their corresponding concatenated test results. It is observed that the error is high during the initial phases of cross validation when the model is not trained well. Towards the last folds, the error reduced drastically since the model is trained well by then with enough number of samples.

B. Hyper parameters for Ridge Regression[19]

The hyper-parameter in case of ridge regression is λ Larger the value of λ, larger will be the regularization which makes the flexibility of the fit very strict.On the other hand, if the value of λ is smaller, higher would be the magnitude of the coefficients.We used the toolbox sklearn to find the optimal value of λ. The toolbox takes in a range of λ values and undergoes a grid search parameter tuning and outputs the optimum value for λ which is found to be 1.5 in our case.

C. Hyper parameters for Neural Networks[19]

Since the ANN is being used as a regressor, the number of nodes in the output layer is 1. The number of nodes in the input layer is equal to the number of features to be trained (in this case 20).

1) Hidden Layer: Initially, we kept the number of neurons between the number of inputs and number of outputs. Further we also tested the network for higher number of neurons in the hidden layer. Finally we set the number of neurons to 200 as it gave the optimum result.

2) Batch size: The gradient descent method operates on a small batch which is a fraction of the training data. There is a trade off between the batch size and the computational complexity of the Network. We have chosen the batch size as 100 for our network.

3) Epochs: An epoch refers to one forward pass and one backward pass of all the training examples. In our case, the model uses a single epoch.

4) Learning Rate: The learning rate is chosen automatically by the RMSProp optimizer in Keras package.

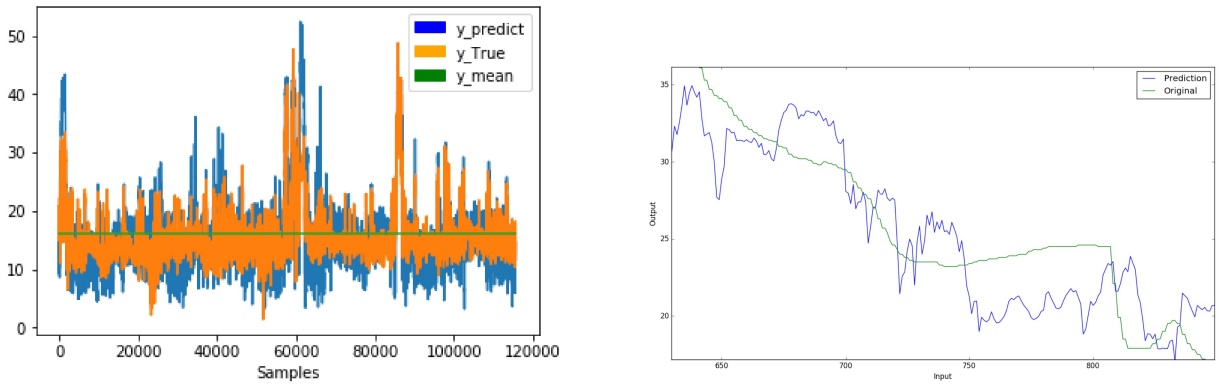
5) Activation function: Since the value of some of the features are not in order of same magnitudes, the sigmoid activation function can cause the problem of vanishing or exploding gradients. Hence, we use the ReLU (Rectified Linear Unit) activation function which not only solves this problem but also makes the computation efficient.

V. Test Cases

A. Test Cases for Regression

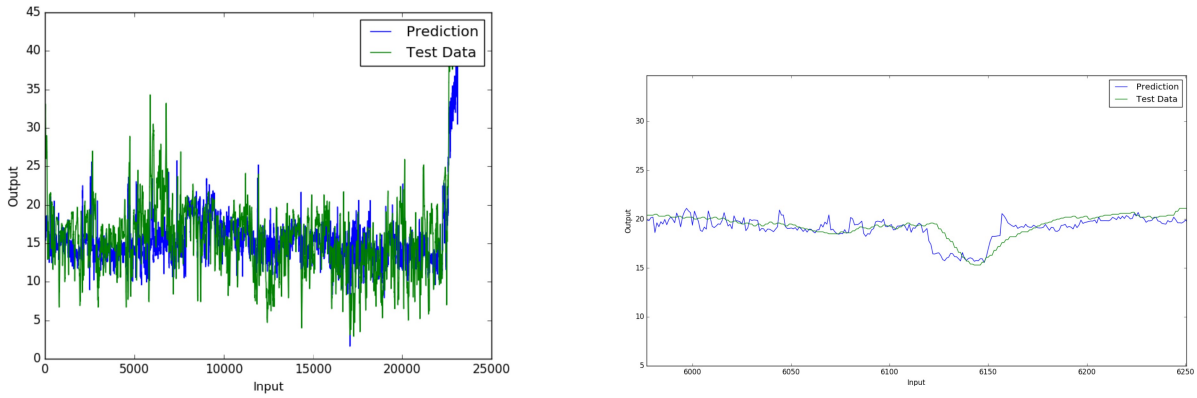
First we took the entire 1,15,618 samples and divided them into 80 percentage as training data and 20 percentage as test data.

1) Cross-validation approach is implemented on the training data with number of folds = 10 and expected prediction error



(a) Over all Samples (b) Zoomed

Fig. 3. Ridge Reg: Test Case 1: Expected Value, True Value and Mean VS Time Samples.



(a) Over all Samples (b) Zoomed

Fig. 4. Ridge Reg: Test Case 2: Prediction Value, True Value VS Time Samples

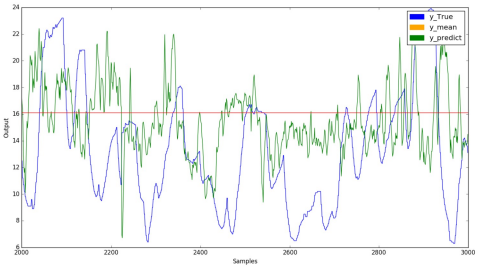


Fig. 5. Neural Network: Prediction Value, True Value and Mean VS Time Samples

is found out. The plot is shown in figure 4. The tabular result is shown in Fig. 7.

2) Test data is given into this trained model after cross-validation resulting in the prediction error for test data. The plot is shown in figure 5.The tabular result is shown in Fig. 8.

Then, we took the whole data together and carried out the below test

1) Again the model is trained with the total 115618 samples with cross-validation approach and the expected error is found out. The plot is shown in figure 6. The tabular result is shown in Fig. 9.

B. Test Cases for ANN

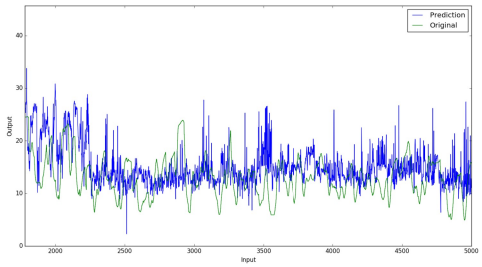
For the purpose of testing, we divided the entire data into training and testing parts. The training data consisted of 80 percent of the data while the remaining 20 percent is saved for testing. The training data is divided into train and cross-validation set. We now train the model on this new training data set and validate it on the cross validation set. The model is tweaked for different values of hyper parameter. We varied the number of neurons in the hidden layer and obtained the RMSE and R-Squared error. This is shown in Fig 10.

Fig. 6 shows the performance of neural network with hidden layer size as 200.

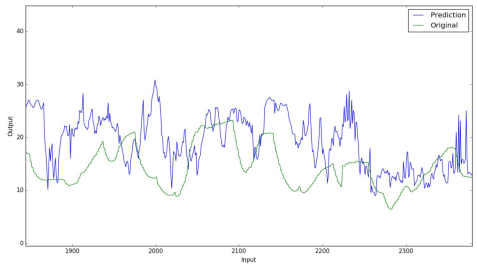
Now using this value as the hidden layer size, we train a Neural Network on the training data(which is 80 percent of the entire data) and find its error on the testing set.

C. Toolboxes

The entire project is implemented in Python 3.6
The python packages we used are:-
- numpy [13](for numerical computations)



(a) Over all Samples



(b) Zoomed

Fig. 6. Neural Network: Prediction Value, True Value VS Time Samples

Iteration for K	MSE	RMSE	R2
1	38.7963651673	6.22867282551287	0.599500883317
2	13.4781777262	3.67126377780147	-0.0751153038527
3	27.2089869183	5.21622343446984	-0.41310077046
4	13.6171546309	3.69014290115513	-0.674883328237
5	12.9442544067	3.59781244739912	-0.449835588765
6	97.9706714133	9.89801350844394	0.431739123513
7	14.7851495199	3.84514622867687	0.235488742427
8	12.8432493722	3.58374795042355	0.298834289326
9	16.6084015569	4.07534066759161	0.0443558670573
10	26.1704765796	5.11570880519651	-0.193589808226

K-fold	MSE	RMSE	R2
Expected Error	27.4422	4.8922	-0.0196

Fig. 7. Ridge Reg: Table: MSE, RMSE and R2 Test Case 1

Test data	MSE	RMSE	R2
Prediction Error	24.049439736365098	4.90402281157	-0.430086442615

Fig. 8. Ridge Reg: Table: MSE, RMSE and R2 Test Case 2

Test data	MSE	RMSE	R2
Expected Error	24.166	4.916	0.47

Fig. 9. Ridge Reg: Table: MSE, RMSE and R2 Test Case 3

- matplotlib[14] (for graph plotting)
- scikit-learn[9] (for implementing regression model)
- pandas[15] (to handle data)
- keras[7] with tensorflow backend (for Neural Networks)
- tensorflow[8] (for optimization)

Number of Neurons in Hidden Layer	MSE	RMSE	R2
200	28.62	5.35	-0.31
220	28.51	5.34	-0.34
250	30.01	5.49	-0.39
150	26.88	5.18	-0.35
130	29.05	5.39	-0.36
40	28.72	5.36	-0.47

Fig. 10. Neural Network: Table: MSE, RMSE and R2

VI. Error Metric

A. Root Mean Squared Error (RMSE)[18]

The RMSE is a quadratic score that measures the average magnitude of the error. It is indifferent to the direction of the error. RMSE gives relatively high weight to large errors.

B. Mean Square Error (MSE)[17]

The MSE is the squared difference between the predicted value and the true value. It is more sensitive to outliers as compared to RMSE.

C. R-squared(R2)[16]

R squared score or coefficient of determination is a measure of how well the future samples are likely to be predicted. It is calculated by the following equation:

$$R^2(y_i, \hat{y_i}) = 1 - \frac{U}{V} \tag{2}$$

where,

$$U = \sum_{i=0}^{n-1} (y_i - \hat{y_i})^2 \tag{3}$$

$$V = \sum_{i=0}^{n-1} (y_i - \overline{y_i})^2 \tag{4}$$

$$\overline{y_i} = \frac{1}{n} \sum_{i=0}^{n-1} (y_i) \tag{5}$$

VII. Conclusion

Based on the estimated error values from both the models, it can be concluded that Ridge Regression outperforms Artificial Neural Network for this particular data set. The possible reason for this could be that the Artificial Neural Network model is overfitting the data resulting in high error values.

References

[1] https://stats.stackexchange.com/questions/2691/making-sense-of-principal-component-analysis-eigenvectors-eigenvalues
[2] http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=7820485
[3] https://www.ncss.com/software/ncss/regression-analysis-in-ncss/
[4] https://en.wikipedia.org/wiki/Stepwise_regression
[5] https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4377081/
[6] http://www.sciencedirect.com/science/article/pii/S2212017312002411
[7] https://keras.io/
[8] https://www.tensorflow.org/
[9] http://scikit-learn.org/stable/
[10] https://medium.com/human-in-a-machine-world/mae-and-rmse-which-metric-is-better-e60ac3bde13d
[11] A. E. Hoerl and R. W. Kennard, Ridge regression: Biased estimation for nonorthogonal problems, Technometrics, vol. 12, no. 1, Feb. 1970.
[12] http://www.real-statistics.com/multivariate-statistics/factor-analysis/principal-component-analysis/
[13] https://www.numpy.org/
[14] http://matplotlib.org/
[15] http://pandas.pydata.org/
[16] http://scikit-learn.org/stable/modules/model_evaluation.html#r2-score
[17] https://en.wikipedia.org/wiki/Mean_squared_error
[18] https://en.wikipedia.org/wiki/Root-mean-square_deviation
[19] http://scikit-learn.org/stable/modules/grid_search.html