Condition Monitoring of Hydraulic Systems

Introduction:

This project addresses the condition assessment of a hydraulic test rig based on multi sensor data. After reading the description of the entire dataset and having a look at the data it was a bit difficult for me to understand a few things directly since I am not from this subject domain. Thus, I had to start with some background research and understand how hydraulic test rigs work.

Background Research:

To start with the background research, I read about hydraulic test rigs on the internet. After that I even read the relevant papers given in the dataset description, some of which are also published by the creators of the dataset. Doing the background research and reading the papers published by the creators helped me understand and realize the approach they have taken to work on such a huge dataset with 17 sensor data. In the paper “Condition Monitoring of a Complex Hydraulic System using Multivariate Statistics” [1], they talk about using Pearson’s correlation coefficient and Spearman’s rank correlation coefficient to eliminate features with correlation below a certain threshold and use the remaining features as input to Linear Discriminant Analysis (LDA). They tried using different classification algorithms like Artificial Neural Networks and Support Vector Machines also but got the best results for all the targets with LDA because it automatically converts the high dimensional data into a lower dimension and trains the model on a lower set of features while still retaining the importance of all the features. Reading the paper on “Automatic feature extraction and selection for classification of cyclical time series data” [2] also helped me in understanding various methods and techniques that can be used for Dimensionality Reduction, Feature Extraction and Feature Selection.

Data Dimensionality:

After doing the required background research, I decided my approach to work on the dataset. I started with exploring each sensor data in more detail and I realized that data points for each cycle of a sensor do not change immediately. I also realized that the duration of each cycle for all the sensors is of 60 seconds but the sampling rates for all the sensors are different which means that each sensor captures data based on its sampling rate. For example, the pressure sensors and the motor power sensor have a sampling rate of 100 Hz which means that those sensors capture 100 data points in every second and 6000 data points in each cycle. The Volume flow sensor has a sampling rate of 10 Hz which means that those sensors capture 10 data points per second and 600 data points in each cycle. All the remaining sensors have a sampling rate of 1 Hz and thus they only capture 1 data point per second and 60 data points in each cycle. Since the data points of each sensor in each cycle did not change immediately I thought of aggregating the data points in each cycle over a small interval of time. I decided to take an average of all the data points in each cycle for all the sensors over 10 seconds but since the sampling rates for all the sensors are different I had to take an average of a different number of data points for sensors with different sampling rates. For sensors with sampling rate of 100 Hz, in 10 seconds 1000 data points would be captured and thus I took an average of every 1000 data points for each cycle. For the sensors with sampling rate of 10 Hz, in 10 seconds 100 data points would be captured and thus I took an average of every 100 data points for each cycle. And similarly, for the sensors with sampling rate of 1 Hz, in 10 seconds 10 data points would be captures and thus I took an average of every 10 data points for each cycle. I decided to take an average over 10 seconds of time to compensate for the difference in sampling rates for all the sensors and also to try to reduce the dimensionality of data by reducing the number of data points for each cycle to 6 for all the sensors.

Data Pre-Processing:

After taking an average of sensor data over 10 seconds for each cycle I combined all the sensor data together in one data frame to have 102 columns representing sensor data (6 data points for each sensor) and 2205 rows representing 2205 cycles for all the sensors together. Before going ahead with any kind of analysis it was really important to scale the data. Scaling data is one of the most important requirements and an important data pre-processing step for the optimal performance of most machine learning algorithms. For scaling the data, I used StandardScaler package of Scikit learn. It rescales the features in such a way that they have the properties of a standard normal distribution with a mean of zero and standard deviation of one. Scaling the features was also important because my immediate next step was to implement Principal Component Analysis (PCA) to further reduce the dimensionality of the dataset. In PCA we are interested in the components that maximize the variance and thus it is an important requirement to scale the features before we do PCA to maximize the variance of the components. It has also been observed in a lot of previous research that PCA with scaled data always gives a better result in a classification algorithm than PCA without scaled data.

Principal Component Analysis:

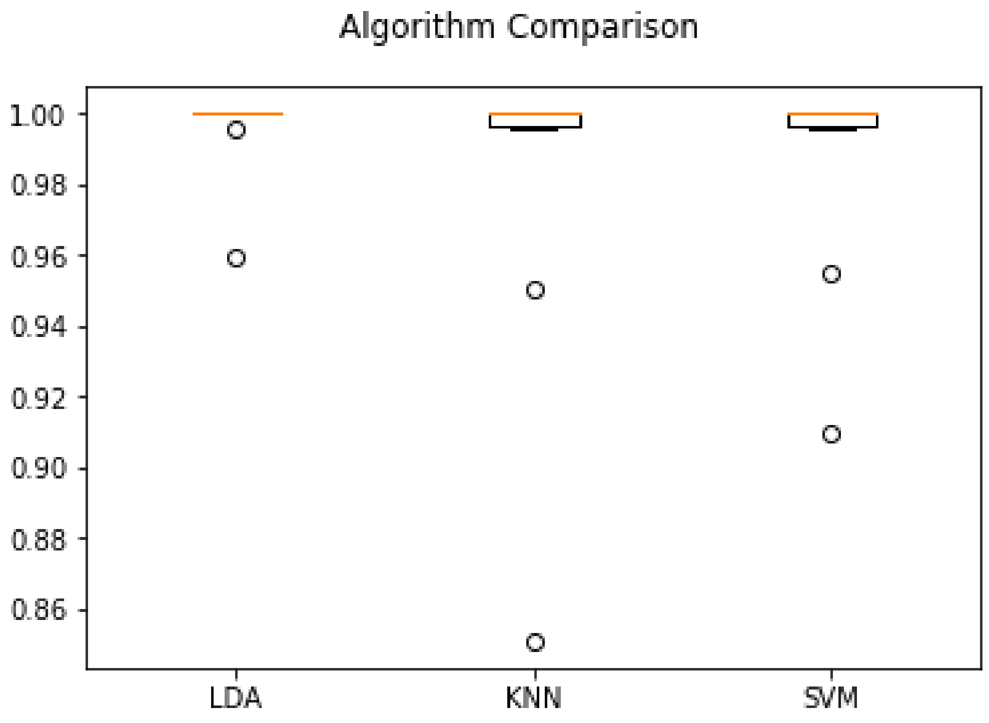
The sensor data is a time series data and after reading the paper on "Automatic feature extraction and selection for classification of cyclical time series data" [2], I realized that Principal Component Analysis is the best way to catch the general trend of a cycle. I decided to do PCA for multiple number of reasons. Firstly, the principal components that we get through PCA are the best linear transformation of the dataset. They also capture the maximum amount of information from all the features that are provided. The first principal components are formed to explain maximum variance while the later principal components capture details and noise. Also, while capturing the general trend of time series data really well it also helps us in reducing the dimensionality of the data. It reduces the input dimension into the number of components specified and speeds up the machine learning algorithm. For all the above-mentioned reasons I decided to go ahead and implement Principal Component Analysis on the scaled data.

To retain maximum information from the dataset I decided to check the minimum number of components that I would need to have the explained variance as 99%. According to the results, minimum 9 components were supposed to be required to have an explained variance of 99% amongst the components. Thus, I implemented PCA with 9 components as the input which helped me to further reduce the dimensionality of data from 102 columns to just 9 columns (one for each component). Thus, decreasing the computational time and reducing the dimensionality of data while ensuring the importance of each feature was the main reason I decided to implement Principal Component Analysis before applying any Classification Algorithm.

Classification and Results:

After implementing PCA it was important for me to train my classifier using the 9 principal components obtained and thus it was important for me to select a correct classifier that would work in this case. Firstly, for the target "Cooler Condition" there were 3 classes and thus it was multi-class classification. Although Logistic regression can do multi-class classification but traditionally it is a method that is used for only binary classification and there are better algorithms that can do multi-class classification than Logistic regression and thus I eliminated that option immediately. Also, after implementing PCA, the resulting dataset having 9 components with just 2205 observations was not a complex dataset to handle and thus I wanted to avoid using any kind of Tree algorithm. I finally decided to select Linear Discriminant Analysis, Support Vector Machines and K- Nearest Neighbours as classification algorithms. I selected Linear Discriminant Analysis especially because it would maximize the separation between the different classes while minimizing the scatter within a class. K nearest neighbours works poorly when the dimension of the data is very high but using PCA we have already reduced the dimensionality of the dataset and thus in that case K nearest neighbours is a good choice to handle multi-class classification. K fold Cross validation was used with 10 splits to divide the dataset into training and testing and to avoid overfitting the model. Model selection package of Scikit learn was used to train the classifiers and also do K Fold Cross Validation. There was not a lot of difference in the result of all the 3 classifiers that were used. The result of the 3 classifiers is given in the table below. The results of all the 3 classifiers were also visualized and compared using box-plots as shown below.

Classifier	Accuracy
Linear Discriminant Analysis	99.5475%
K Nearest Neighbours	97.9636%
Support Vector Machines	98.5950%



References:

[1] Nikolai Helwig, Eliseo Pignanelli, Andreas SchÃ¼tze, â€˜Condition Monitoring of a Complex Hydraulic System Using Multivariate Statisticsâ€™, in Proc. I2MTC-2015 - 2015 IEEE International Instrumentation and Measurement Technology Conference, paper PPS1-39, Pisa, Italy, May 11-14, 2015, doi: 10.1109/I2MTC.2015.7151267.

[2] Tizian Schneider, Nikolai Helwig, Andreas SchÃ¼tze, â€˜Automatic feature extraction and selection for classification of cyclical time series dataâ€™, tm - Technisches Messen (2017), 84(3), 198â€“206, doi: 10.1515/teme-2016-0072.