CMP3749M Big Data Assessment Report

A logo of a university

Description automatically generated

Mathews Joy

[25186202@students.lincoln.ac.uk](mailto:25186202@students.lincoln.ac.uk)

JOY20779936

University of Lincoln

School of Computer Science

When finished when ever I mention a specific function e.g. “toPandas()” include a reference next to it of the link to that functions documentation.

**Task 1 – Analysis of Nuclear Plants dataset**

**Part 1**

In the dataset we analysed for task 1 ”nuclear\_plants\_small\_dataset.csv” we used the pyspark “.isNull()” function to confirm there was no missing data points in the dataset. This function “is used to check if the current expression is NULL/None or column contains a NULL/None value”1. In our scenario we identified that there was no missing values in our dataset, as a result no further action was required. In the case a column contained 1 or more empty values, it is important to identify it and take necessary steps to fix the missing values, this is to avoid skewing the results from analysis you might perform. First it is important to identify what type of missing data you have; “Missing completely at random (MCAR) data is just randomly missing, missing at random (MAR) meaning missing conditionally at random based on another observation or missing not at random (MNAR) which is missing as part of how it is collected (deliberately missing)”2. Based on the previous statement by identifying these types helps in choosing the correct method to handle the missing data. For instance, if the data is classified as MCAR, it might be best to simply exclude those cases from your dataset. Although simply deleting data is not recommend, but as the missing data is MCAR, removing the data would not bias the analysis performed. This is because the missingness is not dependent on observed or unobserved data. But if it’s MAR or MNAR, more sophisticated techniques might be needed to avoid bias in your analysis. These could include imputation, which involves filling in missing values with substituted values such as mean of the column with missing data. Other options include acceptance/ignore them, “this is the most conservative option involves accepting your missing data: you simply leave these cells blank. It’s best to do this when you believe you’re dealing with MCAR or MAR values”3. Although the data seemed quite clean and structured, upon inspection with the “.printSchema()” function (which displays the data frame columns and there datatypes), we can see the column names are inconsistent. For example, in the pressure sensor column names, they have a random space value in the column names, which the other columns do no have. Due to this inconsistent it is best to keep the column names in uniform so we used the “.replace()” function on each column name to check for random empty spaces in the column names and get remove them. This means in future analysis no unexpected errors will be through when trying to access specific columns.

**Part 2**

For this task we are asked to find the summary statistics (minimum, maximum, mean, and median) for each group of subjects (normal and abnormal), and then for each group plot the boxplots for each feature. We can first approach this task by converting the pyspark data frame to “pandas” data frame; “pandas is a fast, powerful, flexible and easy to use open source data analysis and manipulation tool” 4. The reason we convert to pandas is it allows cleaner syntax for performing aggregate functions in general and integrates with graphical visualisations easier, however these calculations can also be performed directly on the pyspark data frame. We can convert our current data frame to pandas via the “.toPandas()” function. Next, we can create a for loop that loop through the 2 statuses (normal and abornal) and finds the min,max mean and median using the “.describe()” function.

A computer screen shot of text

Description automatically generated

We then get the below result. Some key findings we can pick out are, the pressure sensors overall have the highest max values for both normal and abnormal status e.g. (56 for normal and 67 for abnormal). The mean values are all around the similar levels for all features across both normal and abnormal status, one exception being “vibration\_sensor\_3” (normal) displaying a higher overall mean of 19.

A screenshot of a computer

Description automatically generated

The next part of this task revolved around creating boxplots for all features for both Status groups. Benefit of using a boxplot is they are “very effective and easy to read, as they can summarize data from multiple sources and display the results in a single graph” 8. We initially format the data frame so that we can pass it into a seaborn “catplot which is a figure-level interface, which is a more flexible tool that can be used to draw various types of categorical plots, including boxplot” 9. We use the catplot and set the y axis as the features values and the x axis for the feature labels.

A screen shot of a computer program

Description automatically generated

We are then presented the chart in figure 4. Some key takeaways from figure 4 are we can see for 9 of the 12 features there are an abundance of outliers as indicated by the diamond points above and below the whiskers of the boxplots. This is something to consider as we might want to potentially remove these values as it could affect any further analysis or choose to keep them as they can provide valuable insight into the data. We can also highlight vibration sensors tend to have a greater range of data point compared to power and pressure sensors, only exception being pressure sensor 1.

A graph of different colored squares

Description automatically generated

Figure 4. Multi Boxplot comparing all features grouped by status.

**Part 3**

When you produce a correlation matrix of some data, you can identify highly correlated features by looking for pairs of features that have a high correlation coefficient. “A correlation coefficient of 1 indicates a perfect positive correlation, while a correlation coefficient of -1 indicates a perfect negative correlation. A correlation coefficient of 0 indicates no correlation between the two features” 5.

Highly correlated features can be problematic for further processing, such as data classification, because they can introduce multicollinearity, which can lead to overfitting and unstable models. One way to deal with highly correlated features is to remove one of the features from the dataset. Another way is to use feature selection algorithms to select a subset of features that are most relevant to the classification task. Principal component analysis (PCA) can also be used to reduce the dimensionality of the dataset by transforming the correlated features into a set of uncorrelated features 6.

In our scenario we have constructed a correlation matrix using the seaborn visualisation package and the “.heatmap()” function. We pass in our data frame for the input to the “.heatmap” function ensuring we drop the “Status” column as this is binary text column and would bring much value, and then applying the “.corr()” function. “.corr() calculates the correlation matrix whose elements range is [-1, 1], by default it uses Pearson Correlation coefficient. sns.heatmap is just a way to display using colours how strong the correlations are, where a lighter colour cream indicates a positive correlation and a stronger darker colour black highlights a negative colleration.”7.

After observing the correlation matrix, we can see that there are some features with high correlation with one another for example “Power\_range\_sensor\_4” and “Pressure\_sensor\_4” with a correlation of 0.82. In general features with high correlation like our example can increase the complexity of the model, and will pose no benefit, so it could be a beneficial idea to remove one of the features. On the contrary, features such as “Power\_sensor\_3” and “Pressure\_sensor\_1” have a correlation of fill in number, this is a negative correlation and shows as one feature increases the other slightly decreases. So this pair of features for example should be kept in the data frame for further processing and data insights, as they will bring a less likely chance of multicollinearity and overfitting.

**Task 2 - MapReduce for Margie Travel dataset**

**Task 3 - Big Data Tools and Technology Appraisal**

**References**

1. [PySpark isNull() & isNotNull() - Spark By {Examples} (sparkbyexamples.com)](https://sparkbyexamples.com/pyspark/pyspark-isnull/)
2. Week 3 big data workshop
3. [Missing Data | Types, Explanation, & Imputation (scribbr.co.uk)](https://www.scribbr.co.uk/stats/missing-values/)
4. <https://pandas.pydata.org/>
5. [How to Read a Correlation Matrix - Statology](https://www.statology.org/how-to-read-a-correlation-matrix/)
6. [linear regression - What we should do with highly correlated features? - Stack Overflow](https://stackoverflow.com/questions/65302136/what-we-should-do-with-highly-correlated-features)
7. <https://stackoverflow.com/questions/61281035/what-does-the-corr-method-do-in-pandas-and-how-does-it-relate-it-to-the-heatm>
8. <https://asq.org/quality-resources/box-whisker-plot>
9. <https://seaborn.pydata.org/generated/seaborn.catplot.html>

GO through all screenshots and add figure subtitles with quick description maybe