Bosch Manufacturing Line Failure Analysis

Milestone 1: Project Proposal

Project Statement:

Predict or catch internal failures of components in a Bosch production line so as to manufacture reliable products, lower manufacturing re-work and product recalls for the end customer.

File source:

https://www.kaggle.com/c/bosch-production-line-performance/data

Motivation:

It is a common problem across manufacturing companies. Quality control, conducts random checks on unfinished and finished goods. Many times, defects in unfinished goods are not caught in time. They are found later in the finished stage causing re-work and lost sales.

This project will give us a sense on which production line or machine has the most defect rate and which should be fixed for better productivity.

Example: In a glass manufacturing line, control room had to discard a batch of glass, when they found a surface crack on the glass ribbon or big spots. This use to happen 2 to 3 times a day and sometimes due to unknown reasons.

Data Story:

More than 14 GB data split between 7 GB train and 7 GB test.

Contains:

- 4 production lines L0 to L3
- 52+ processing stations S0 to S51
- 4000+ features F0 to F4262
- 1,183,747 rows in each training table (x 3) to be merged.
- System Use: Spark Dataframe and/or postgres (to be explored) to find trends and chart it using pandas.

Project Goals: Interoperate between various systems and apply our learnings. Example:

- If we load this data set in postgres, we will use database and SQL skills from 511, 611.
- Since, this analysis is dealing with 'Volume', we will get to think about efficient data processing 515 and big data spark dataframes from 516. This will give us the foundation to work on other projects requiring 'Velocity' and 'Variety' in addition to 'Volume'
- Referring to column fields and data values, we feel 505 data wrangling and 591 data mining concepts will help us find useful trends in data example: similarity, dissimilarities, eigen decomposition.
- Above should help us extract trends and visualize in pandas using matplotlib, altair and show uncertainty in defect rate per production line. (%Passed/total Vs %Failed/total).

- Lastly, think about bias in data and try relate it to ethics. Example. Looking at number of counts 99% features passed the inspection and leaving only 1% failed. This set is highly imbalanced. So, it will be interesting to think about how to handle classification bias which machine can produce, without human intervention.

```
spark.sql(query_num).show()
+--------+----------------+-------+
|Response|count(Response)|percent|
+--------+----------------+-------+
|       0|         1176868|   0.99|
|       1|            6879|   0.01|
+--------+----------------+-------+
```

Challenges:

- **Large number of Rows and columns:** Not sure how to consolidate 3 tables in to one big file.
  - Example: Should we load it in postgres first, and then process it in spark?
  - How should we load 1+ million rows and 4000+ columns per table and find common relationships?
    - **Based on initial exploration:**
      Production line and Station is the only common thread. Features seems to vary across each category, dates and numeric tables. Example: category feature 25 represented by F25, has a date D26 when this feature was timestamped and has numeric response F24 under which the quality inspection either passed or failed.
- **Interpretation of Results & Charting:** Rough idea. Once, above relationship is established for all production lines, we should be able to see failure counts and assess which conditions are causing it to break as shown in the figure.
    - Example: we can calculate pass and fail percentage per production line and see where we would need most maintenance.

```
In [62]: #Maximum pass and Failure Counts:
         fail_25.show(1), pass_25.show(1)
+---------+--------+-----+
|L0_S1_F24|Response|count|
+---------+--------+-----+
|   -0.058|       1|   51|
+---------+--------+-----+
only showing top 1 row

+---------+--------+-----+
|L0_S1_F24|Response|count|
+---------+--------+-----+
|   -0.045|       0| 8449|
+---------+--------+-----+
only showing top 1 row
```

- **Merging External Datasets:** It is not possible to merge external files to this dataset due to its unique column names and problem domain. But, it would be possible to fetch more data in future (say bootstrap or use this code for another production line) and see how results would vary.
- **Other notebooks on the internet:** This project was a part of kaggle competition held in 2016. There were 1,370 teams and many notebooks created by other teams which can be found as per the link below:
  https://www.kaggle.com/c/bosch-production-line-performance/notebooks?sortBy=relevance&group=everyone&search=spark&page=1&pageSize=20&competitionId=5357
  However, we realized that our goals and selection of tools differ from other approaches. Example: most teams have used 'python chunking' method to read the data, while we want to approach this from efficient data processing standpoint 515/516. Lastly, we feel above link will be a good resource to see how others have approached this problem and explore what other enhancements we can bring to this work. After we have well connected tables, we will be able to add more.

Team Formation: Team plans to work equally and run codes together so that we collaborate and ensure each member understands how it is done. We are still exploring how to have a common place to code and share work, example git/colab/databricks etc. Not sure yet.

CSV File description:

- train_numeric.csv - the training set numeric features (this file contains the 'Response' variable)
- test_numeric.csv - the test set numeric features (you must predict the 'Response' for these Ids)
- train_categorical.csv - the training set categorical features
- test_categorical.csv - the test set categorical features
- train_date.csv - the training set date features
- test_date.csv - the test set date features
- sample_submission.csv - a sample submission file in the correct format

Big Ideas: Prepare data for ML so that we can predict component failures during product assembly.

Key questions: (Some may be outside of current project scope but worth considering)

- How should modelling approach shift, if data is gathered in batches via IOT sensors and stored in different file formats?
- How to ensure that model findings are accurate and unbiased?
- Who needs to be altered (stakeholder), in what format and how often?
- How to automate data flow from data gathering through to dashboarding?

Data Challenges and Initial Approach:

Challenges: Analyze big data stored in 3 csv files (ALL STRINGS), different columns labels, multiple missing values, containing imbalance in target variable (99% passed Vs 1% failed quality inspection), difficult loading/reshaping and merge tables.
Hence, we require higher compute power (> local machine) and storage for data wrangling and spark analysis.
Approach: We intend to use AWS RDS POSTGRES or AWS S3 SELECT to load datafiles and then run spark analysis. (up for debate)

Execution plan & Resources: Connect AWS RDS to Postgres or AWS SELECT, then load data, connect to pyspark, finally collaborate on COLAB for coding ideas and improve processing efficiency.

(Let's refer to initial proposal for more on project goals and challenges)