CONFIDENTIAL

Machine Breakdown Analytics

→ *Data Analysis & Simple Machine Learning Predictions for Chemical Line*

Content

01 Project Objectives

What is the scope or KPI?

1 Project Objective

- **Machine**

The company wants to be aware of the **top causes of downtime** and its **occurrence frequency/patterns** so as to plan ahead before the downtime occurs.

- **Man**

With efforts to recognize employees' performance, the company wants to reward the **best technician/repairs** over the years but do not know how to measure/judge their performance as they are considered a more technical role.

02 Analyzing Dataset

Understanding the Given Dataset

2-1 Understanding the Dataset

A. Total no. of Rows and Columns

Rows	700
Columns	30

In [2]:
```
# Load dataset
data = pd.read_excel("IL DES Line 4 Breakdown Data.xlsx")
pd.set_option('display.max_columns', None)
data.head()
```

Out[2]:

	ID.No	MC.No	Area	MC.Name	Report Date & Time	Problem Description	M/C Status When Attend	M/C Status During Repair	M/C Status After Attend	No of people attend
0	216422	L108DV	NC-B314	DES 4-Chemical Developing 1	2014-02-20 03:55:15	few rollers with loose 'o' ring causing pnl ov...	SWA	SDR	OK	1.0
1	216864	L108DV	NC-B314	DES 4-Chemical Developing 1	2014-02-26 16:53:06	dev system level sensor data jammed...chemical...	SWA	SDR	OK	1.0
2	219122	L108DV	NC-B314	DES 4-Chemical Developing 1	2014-03-31 05:12:26	filter pipe leaking	NP	NP	OK	1.0
3	220959	L108DV	NC-B314	DES 4-Chemical Developing 1	2014-04-28 20:47:53	air hose leaking	RWA	RDR	OK	1.0
4	222147	L108DV	NC-B314	DES 4-Chemical Developing 1	2014-05-19 14:04:31	air leakage	RWA	RDR	OK	1.0

In [3]:
```
# Check total rows and columns of the dataset
data.shape
```

Out[3]: (700, 30)

CONFIDENTIAL

2-2 Understanding the Dataset

B. Total no. of Machine No.

Total No of Machine No	21

```
In [4]: data['MC.No '].nunique()
Out[4]: 21
```

CONFIDENTIAL

2-3 Understanding the Dataset

C. Null Data

Total Empty Rows	700
Total Empty Data	14

In [5]:
```
# Find out the breakdown of null values in different columns
data.isna().sum()
```

Out[5]:
```
ID.No                                          0
MC.No                                          0
Area                                           0
MC.Name                                        0
Report Date & Time                             0
Problem Description                            0
M/C Status When Attend                         3
M/C Status During Repair                       3
M/C Status After Attend                        0
No of people attend                            3
Action Taken                                   3
Actual Repair Start Time(Attend Date)          6
Troubleshoot Complete Time                     6
Actual Repair Complete Time(Complete Date)     6
Fab/Spare parts sourcing Time(mins)            6
Breakdown Repair Category                      6
KIV Reason                                     6
Parts Required                               700
Qty Required                                   6
Wait AM/PM                                     3
Attend Wait time (mins)                        6
Troubleshoot Hours (mins)                      6
Rectify Hours (mins)                           6
Repair Hours (mins)                            6
Remarks                                      700
Repairer 1                                     6
Repairer 2                                   690
User_Feedback                                700
Feedback By                                  700
Feedback Entry Time                          700
dtype: int64
```

In [6]:
```
# Find out if there's any duplicated entries
data.duplicated().sum()
```

Out[6]: 0

CONFIDENTIAL

2-4 Understanding the Dataset

D. Managing Null Data

Approach 1

Since there are 5 columns that have null values for all 700 columns, therefore we will drop the columns:

- Parts Required
- Remarks
- User_Feedback
- Feedback_By
- Feedback_Entry_Time

In [9]:
```
# Copy original dataset to retain integrity
data1 = data.copy()

# Drop 5 Colmuns with all null values: Parts Required, Remarks, User_Feedback, F
data1 = data1.drop(data1.columns[[17, 24, 27, 28, 29]], axis=1)
data1.head()
```

Out[9]:

	ID.No	MC.No	Area	MC.Name	Report Date & Time	Problem Description	M/C Status When Attend	M/C Status During Repair	M/C Status After Attend	No of people attend
0	216422	L108DV	NC-B314	DES 4-Chemical Developing 1	2014-02-20 03:55:15	few rollers with loose 'o' ring causing pnl ov...	SWA	SDR	OK	1.0
1	216864	L108DV	NC-B314	DES 4-Chemical Developing 1	2014-02-26 16:53:06	dev system level sensor data jammed...chemical...	SWA	SDR	OK	1.0
2	219122	L108DV	NC-B314	DES 4-Chemical Developing 1	2014-03-31 05:12:26	filter pipe leaking	NP	NP	OK	1.0
3	220959	L108DV	NC-B314	DES 4-Chemical Developing 1	2014-04-28 20:47:53	air hose leaking	RWA	RDR	OK	1.0
4	222147	L108DV	NC-B314	DES 4-Chemical Developing 1	2014-05-19 14:04:31	air leakage	RWA	RDR	OK	1.0

2-5 Understanding the Dataset

D. Managing Null Data

Rows	**700**
Columns	**25**

```
In [10]: # Check again for null values after dropping the 5 columns
         data1.isna().sum()

Out[10]: ID.No                                        0
         MC.No                                        0
         Area                                         0
         MC.Name                                      0
         Report Date & Time                           0
         Problem Description                          0
         M/C Status When Attend                       3
         M/C Status During Repair                     3
         M/C Status After Attend                      0
         No of people attend                          3
         Action Taken                                 3
         Actual Repair Start Time(Attend Date)        6
         Troubleshoot Complete Time                   6
         Actual Repair Complete Time(Complete Date)   6
         Fab/Spare parts sourcing Time(mins)          6
         Breakdown Repair Category                    6
         KIV Reason                                   6
         Qty Required                                 6
         Wait AM/PM                                   3
         Attend Wait time (mins)                      6
         Troubleshoot Hours (mins)                    6
         Rectify Hours (mins)                         6
         Repair Hours (mins)                          6
         Repairer 1                                   6
         Repairer 2                                 690
         dtype: int64
```

Result:

Data left with 700 rows and 25 columns

CONFIDENTIAL

2-6 Understanding the Dataset

D. Managing Null Data

Approach 2

We assume that those with null values in 'Repairer 2' Column is because there's only 1 repairer

• Replace the Null Values in 'Repairer 2' Column with "No Repairer 2" (String)

Result:

Left 6 rows with null data

In [11]:
```
# Replace 'Repairer 2' Column null cells with "No Repairer 2"

data1['Repairer 2'] = data1['Repairer 2'].fillna("No Repairer 2")
```

In [12]:
```
# Check again for null values after filling null cells of 'Repairer 2' Column
data1.isna().sum()
```

Out[12]:
```
ID.No                                        0
MC.No                                        0
Area                                         0
MC.Name                                      0
Report Date & Time                           0
Problem Description                          0
M/C Status When Attend                       3
M/C Status During Repair                     3
M/C Status After Attend                      0
No of people attend                          3
Action Taken                                 3
Actual Repair Start Time(Attend Date)        6
Troubleshoot Complete Time                   6
Actual Repair Complete Time(Complete Date)   6
Fab/Spare parts sourcing Time(mins)          6
Breakdown Repair Category                    6
KIV Reason                                   6
Qty Required                                 6
Wait AM/PM                                   3
Attend Wait time (mins)                      6
Troubleshoot Hours (mins)                    6
Rectify Hours (mins)                         6
Repair Hours (mins)                          6
Repairer 1                                   6
Repairer 2                                   0
dtype: int64
```

2-7 Understanding the Dataset

D. Managing Null Data

Approach 3

With 6 rows of the data have multiple null values

- Drop All the 6 Rows with the Null Values

In [13]:

```python
# Display only thos rows with null values
data1[data1.isna().any(axis=1)]
```

Out[13]:

	ID.No	MC.No	Area	MC.Name	Report Date & Time	Problem Description	M/C Status When Attend	M/C Status During Repair	M/C Status After Attend	No of people attend
448	294973	L110DV	NC-B314	DES 4-Chemical Developing 3	2017-10-13 02:21:34	developing make-up tank sensor level not working	NaN	NaN	nil	NaN
494	297310	L118ET	NC-B314	DES 4-Etching Chamber 5	2017-11-27 07:32:57	etching m15 sensor problem	NaN	NaN	nil	NaN
578	306056	L119ET	NC-B314	IL DES 4-Etching Chamber 6	2018-05-25 08:56:17	hci dosing pump haing problem ..hcl keep dosin...	RWA	RDR	OK	1.0
579	306476	L119ET	NC-B314	IL DES 4-Etching Chamber 6	2018-06-04 05:51:54	etching door sensor problem cot not start mc	SWA	SDR	OK	1.0
580	307416	L119ET	NC-B314	IL DES 4-Etching Chamber 6	2018-06-22 11:44:17	re-cycle fecl meter c no data display...	RWA	RDR	OK	1.0
626	302311	L127WA	NC-B314	DES 4-Water Rinse	2018-03-04 01:09:02	water pipe leaking	NaN	NaN	nil	NaN

CONFIDENTIAL

2-8 Understanding the Dataset

D. Managing Null Data

Rows	694
Columns	25

```
In [14]: # Check again for any null values in dataset
         Final_data.isna().sum()

Out[14]: ID.No                                          0
         MC.No                                          0
         Area                                           0
         MC.Name                                        0
         Report Date & Time                             0
         Problem Description                            0
         M/C Status When Attend                         0
         M/C Status During Repair                       0
         M/C Status After Attend                        0
         No of people attend                            0
         Action Taken                                   0
         Actual Repair Start Time(Attend Date)          0
         Troubleshoot Complete Time                     0
         Actual Repair Complete Time(Complete Date)     0
         Fab/Spare parts sourcing Time(mins)            0
         Breakdown Repair Category                      0
         KIV Reason                                     0
         Qty Required                                   0
         Wait AM/PM                                     0
         Attend Wait time (mins)                        0
         Troubleshoot Hours (mins)                      0
         Rectify Hours (mins)                           0
         Repair Hours (mins)                            0
         Repairer 1                                     0
         Repairer 2                                     0
         dtype: int64
```

```
In [15]: # Check processed dataset's total number of rows and columns
         Final_data.shape

Out[15]: (694, 25)
```

Result:

Data left with 694 rows and 25 columns

2-6 Understanding the Dataset

E. Final Data

We will further drop columns which are not useful for EDA or have any correlation in the Final_data

Columns to Drop:
•ID.No
•Area

Rows	694
Columns	25
Total No of Machine No	21

Total Empty Rows	0
Total Empty Data	0

In [18]:
```python
# Further dropping of non-useful columns
Final_data = Final_data.drop(data1.columns[[0, 2]], axis=1)
Final_data.head()
```

Out[18]:

	MC.No	MC.Name	Report Date & Time	Problem Description	M/C Status When Attend	M/C Status During Repair	M/C Status After Attend	No of people attend	Action Taken	R TI
0	L108DV	DES 4-Chemical Developing 1	2014-02-20 03:55:15	few rollers with loose 'o' ring causing pnl ov...	SWA	SDR	OK	1.0	replace some worn out oring @ develoving roller	[illegible]
1	L108DV	DES 4-Chemical Developing 1	2014-02-26 16:53:06	dev system level sensor data jammed...chemical...	SWA	SDR	OK	1.0	clean sensor and reset system.	[illegible]
2	L108DV	DES 4-Chemical Developing 1	2014-03-31 05:12:26	filter pipe leaking	NP	NP	OK	1.0	replace the pvc filter connector.	[illegible]
3	L108DV	DES 4-Chemical Developing 1	2014-04-28 20:47:53	air hose leaking	RWA	RDR	OK	1.0	-trim and re-connect broken airhose.	[illegible]
4	L108DV	DES 4-Chemical Developing 1	2014-05-19 14:04:31	air leakage	RWA	RDR	OK	1.0	replace air connector.	[illegible]

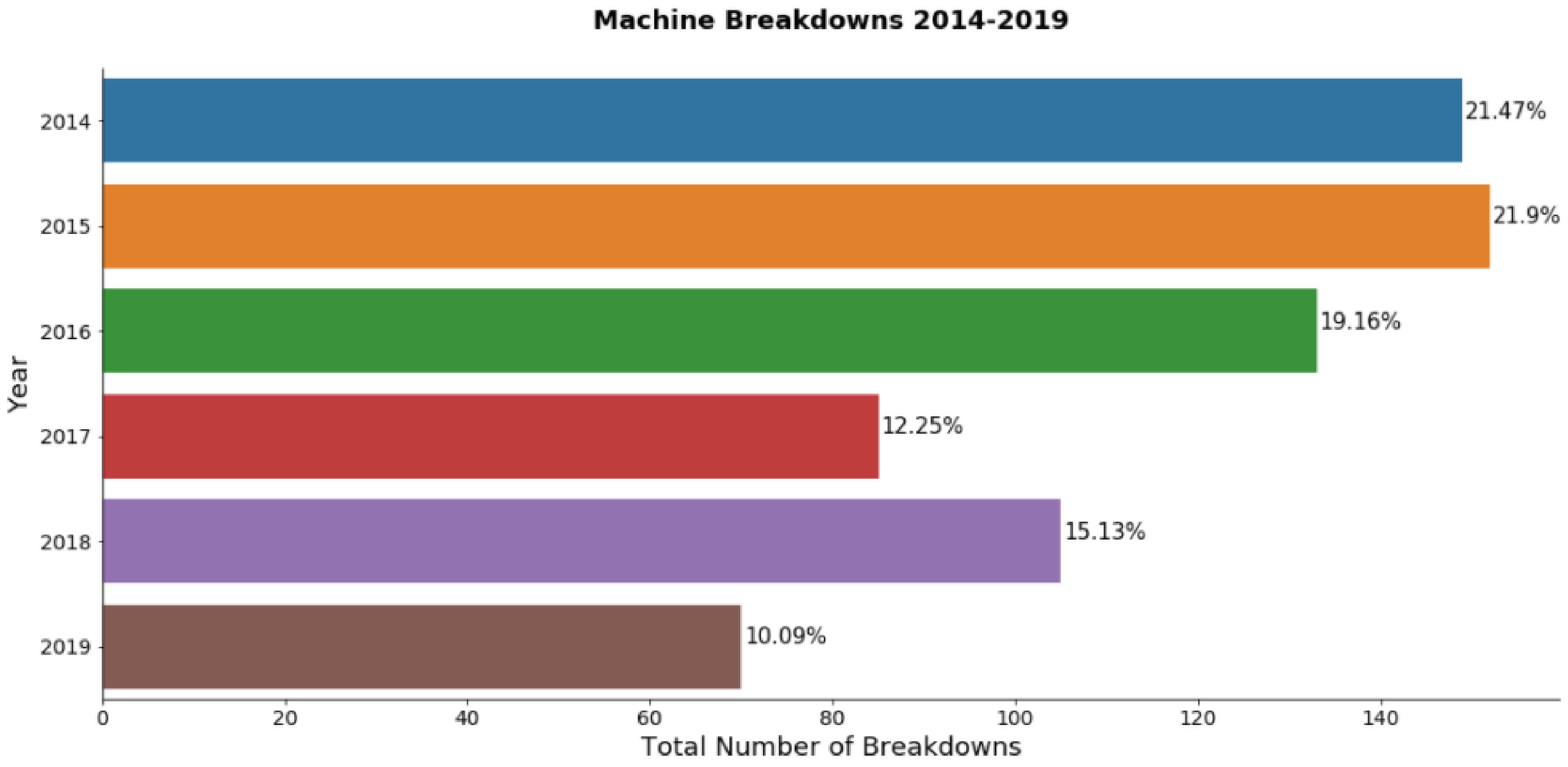
CONFIDENTIAL

03 Analysis of Errors

Study of Breakdowns of Chemical Line

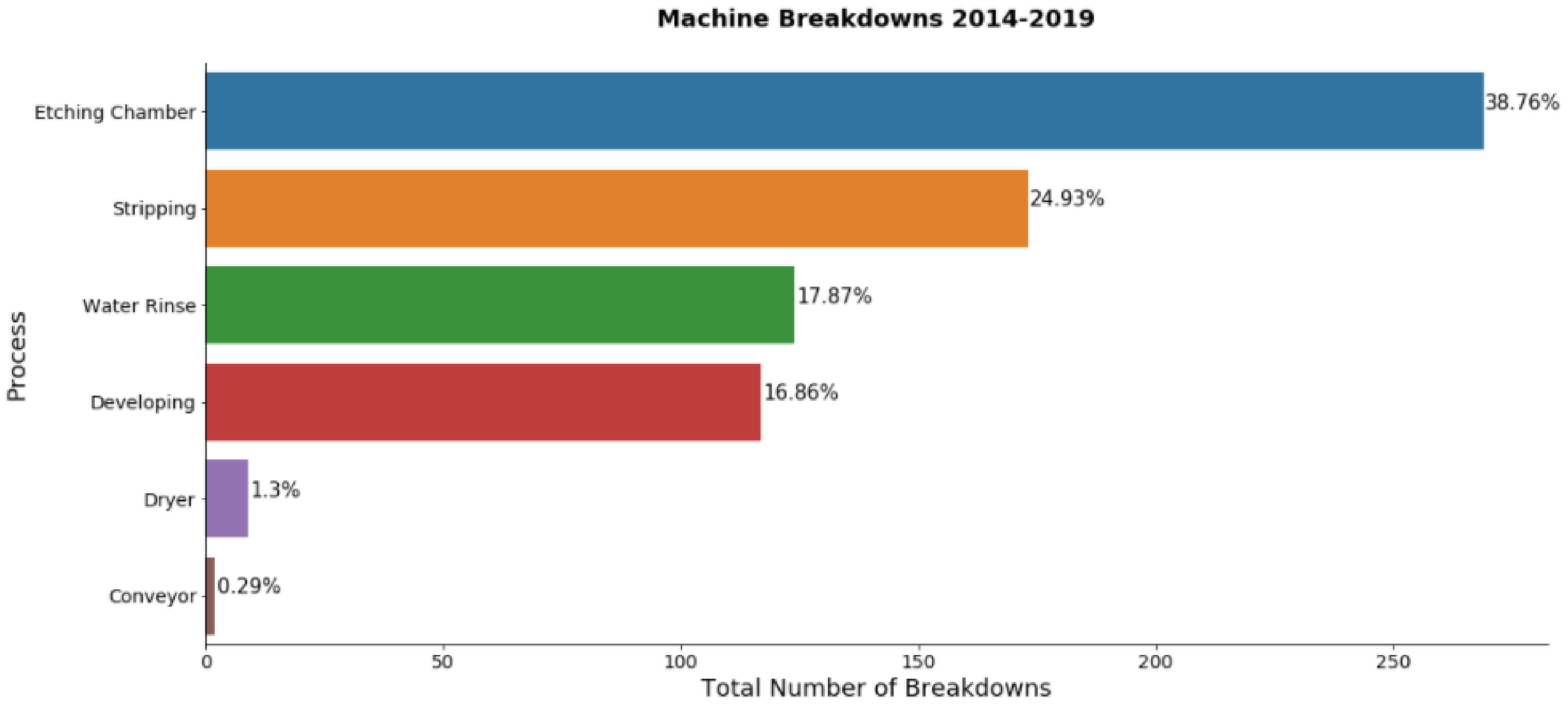
3-1 Breakdown Analysis

A. Yearly Breakdown Distribution Overview



3-2 Breakdown Analysis

B. Process Breakdown Distribution Overview



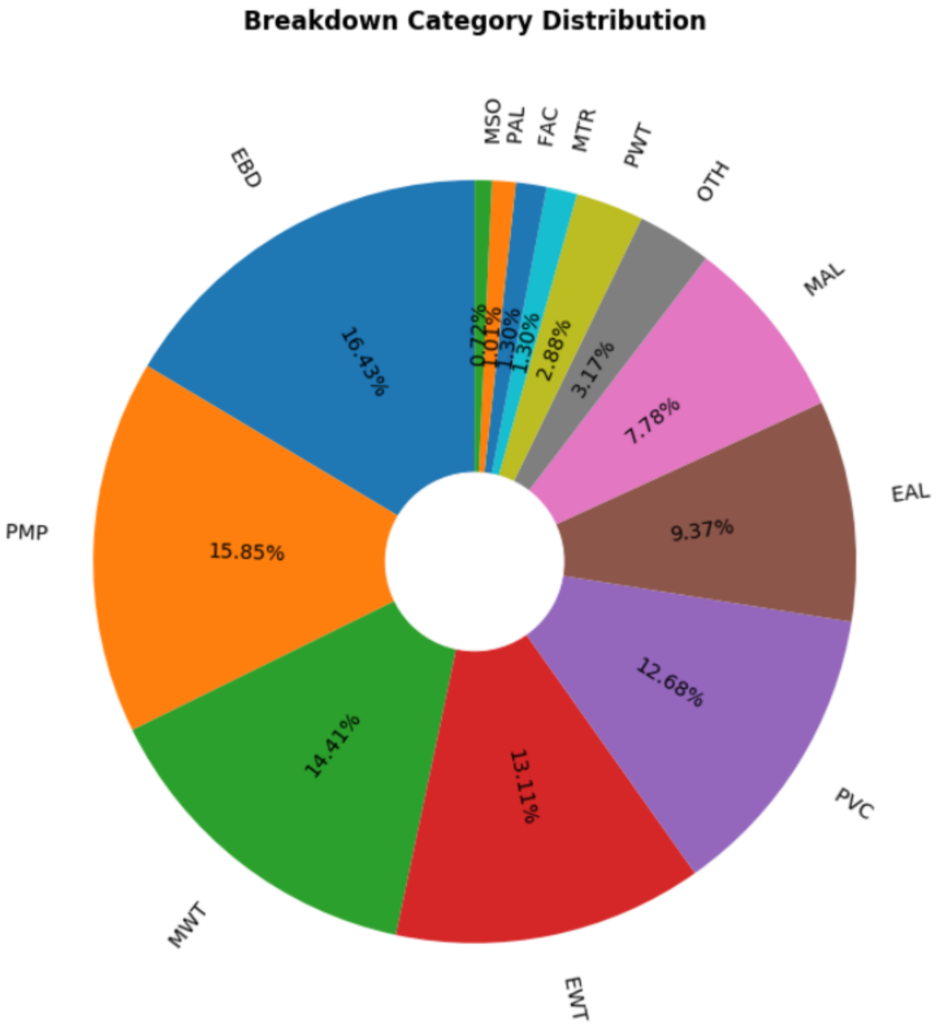
CONFIDENTIAL

3-3 Breakdown Analysis

C. Top Breakdown Distribution

Top 3 Breakdown Repair Categories are:

- Electrical Bit Device (EBD)
- Pump (PMP)
- Main Ware Trip (MWT)



Breakdown Repair Category

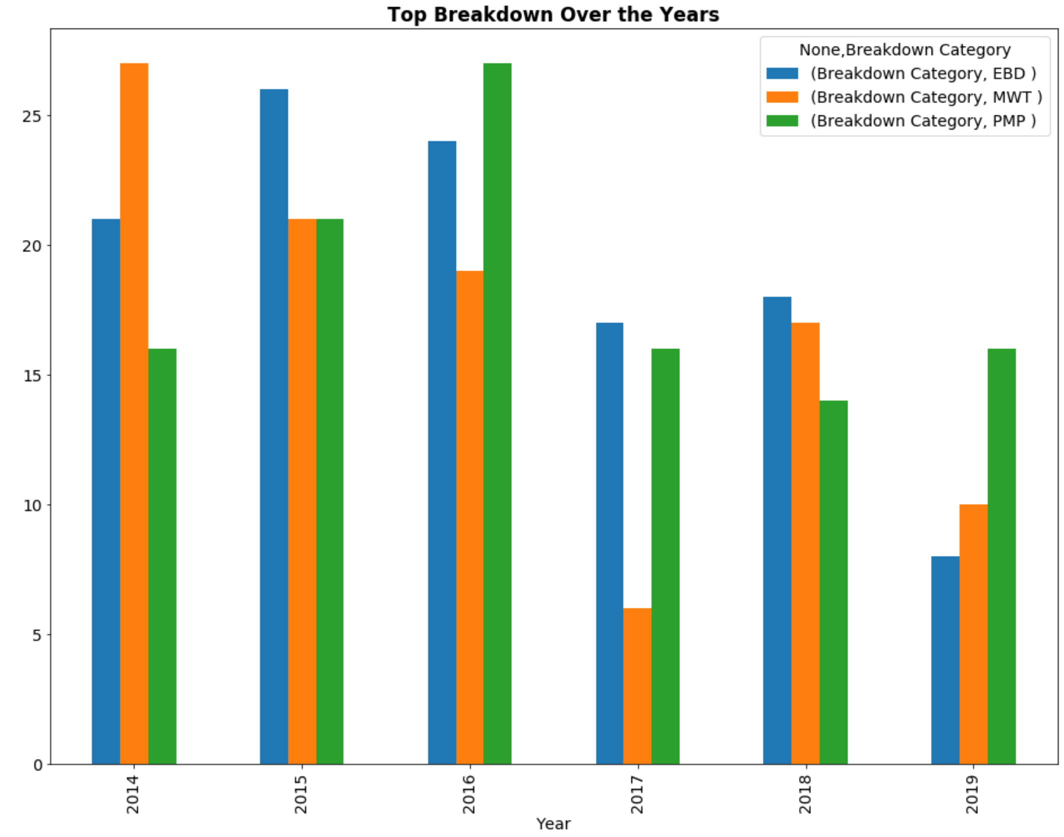
EBD	114
PMP	110
MWT	100
EWT	91
PVC	88
EAL	65
MAL	54
OTH	22
PWT	20
MTR	9
FAC	9
PAL	7
MSO	5

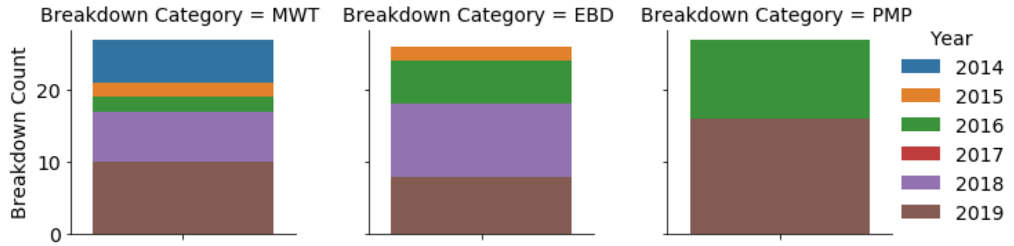
CONFIDENTIAL

3-4 Breakdown Analysis

D. Top 3 Breakdown by Years

- Overall total Top 3 Breakdowns is lower compared to the peak in 2016
- Not much significant pattern can be found by analyzing the Top 3 Breakdown over the Years (2014–2017)



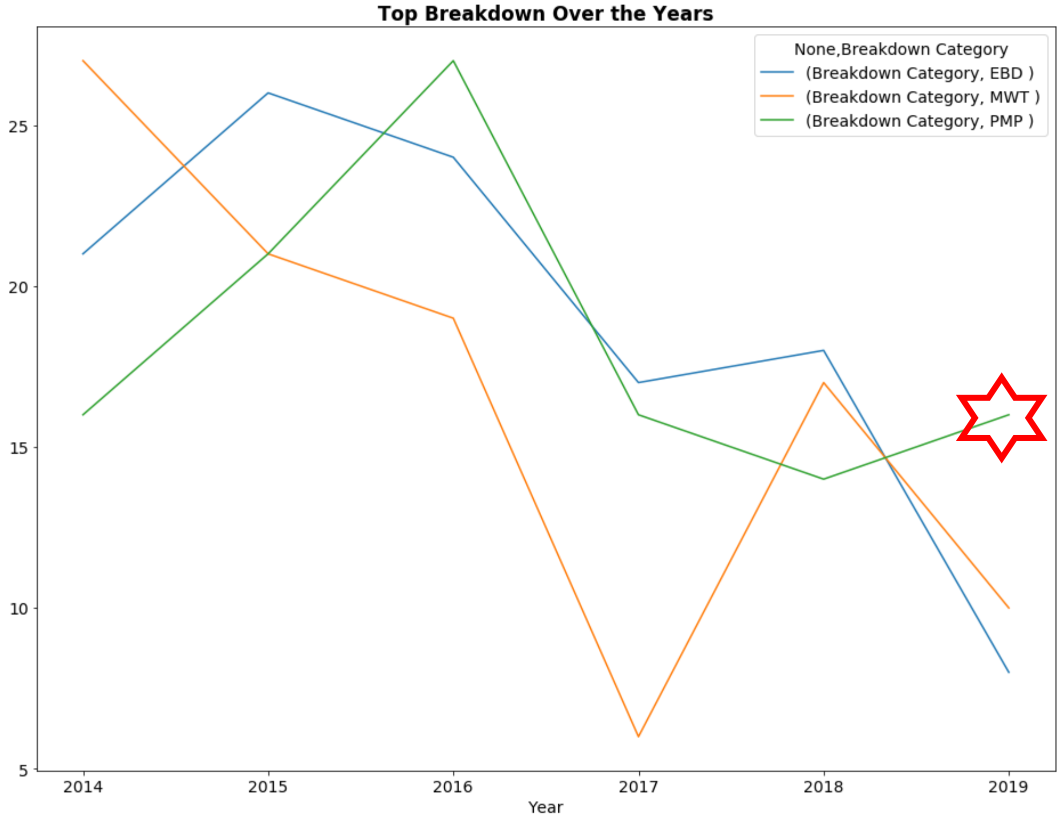


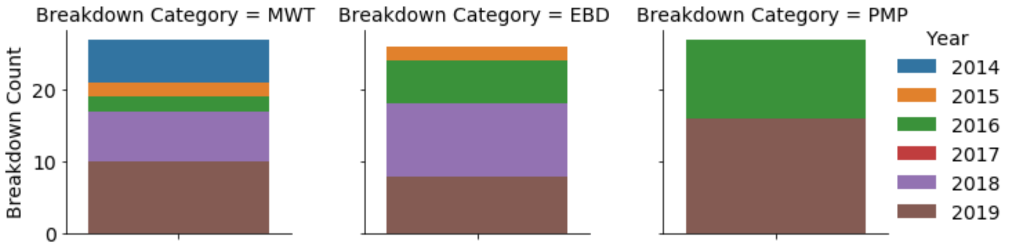
CONFIDENTIAL

3-5 Breakdown Analysis

D. Top 3 Breakdown by Years

- However, an upward trend of the Pump repair is observed.
- There might be a possible pattern of the Pump repair peaks every 2 years (2014 ~ 2016, 2017 ~ 2019)

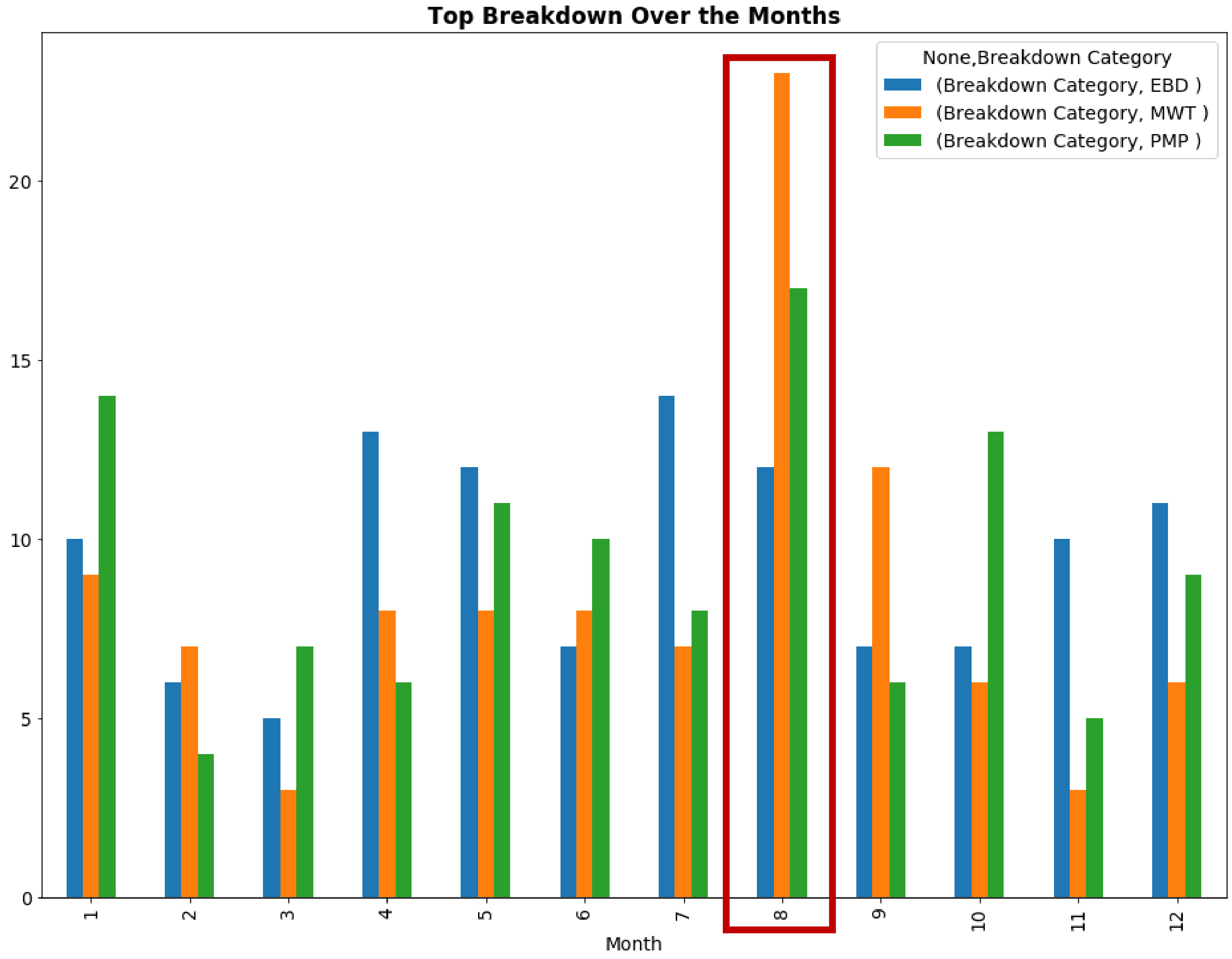




3-6 Breakdown Analysis

E. Top 3 Breakdown by Months

- Overall all Top 3 Breakdowns occurs in all of the months
- There is a spike trend of Main Ware Trip (MWT) and Pump (PMP) repair in August

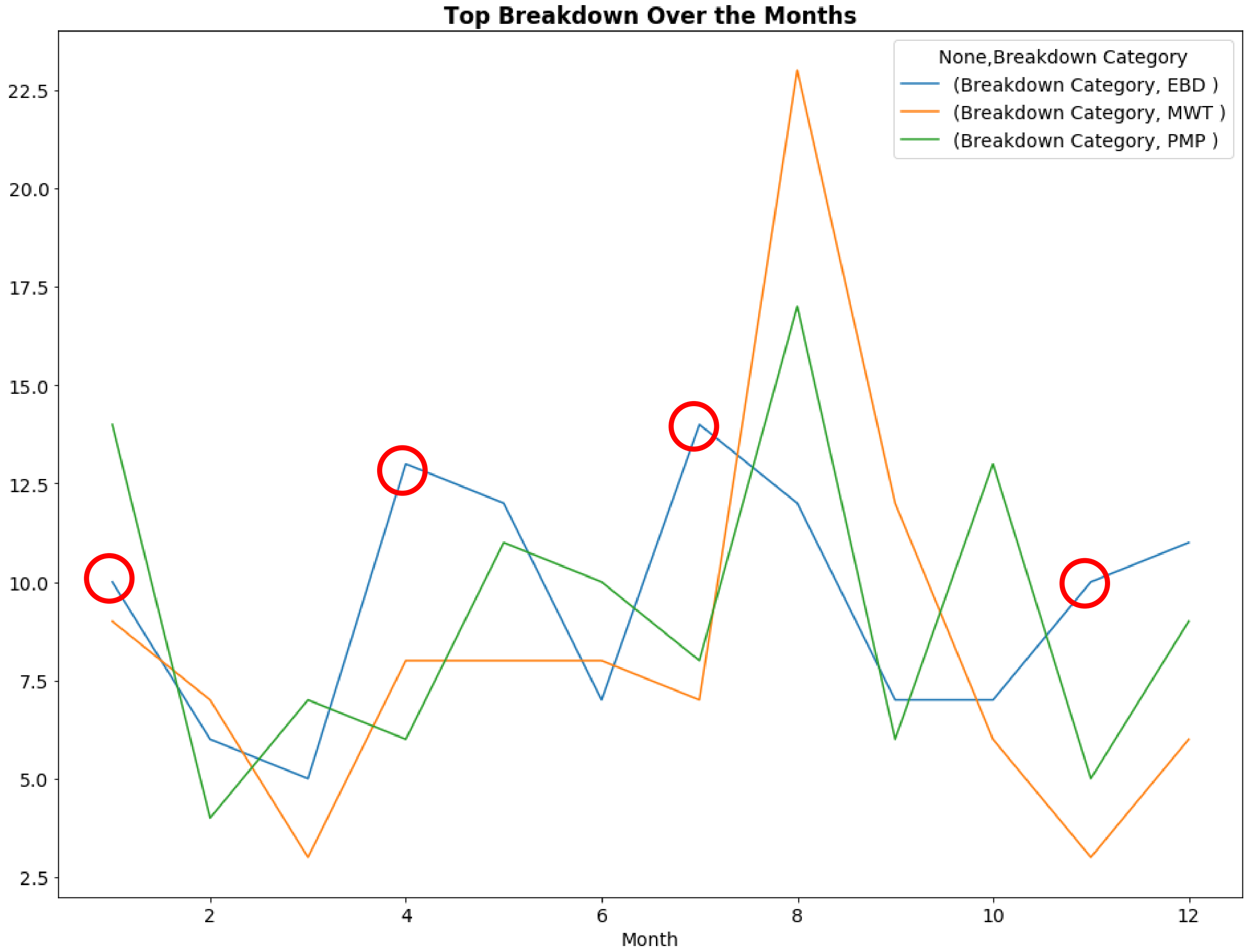


CONFIDENTIAL

3-7 Breakdown Analysis

E. Top 3 Breakdown by Months

- For Electrical Bit Device (EBD) repair, there is a possible trend of increasing in breakdown every 2 months

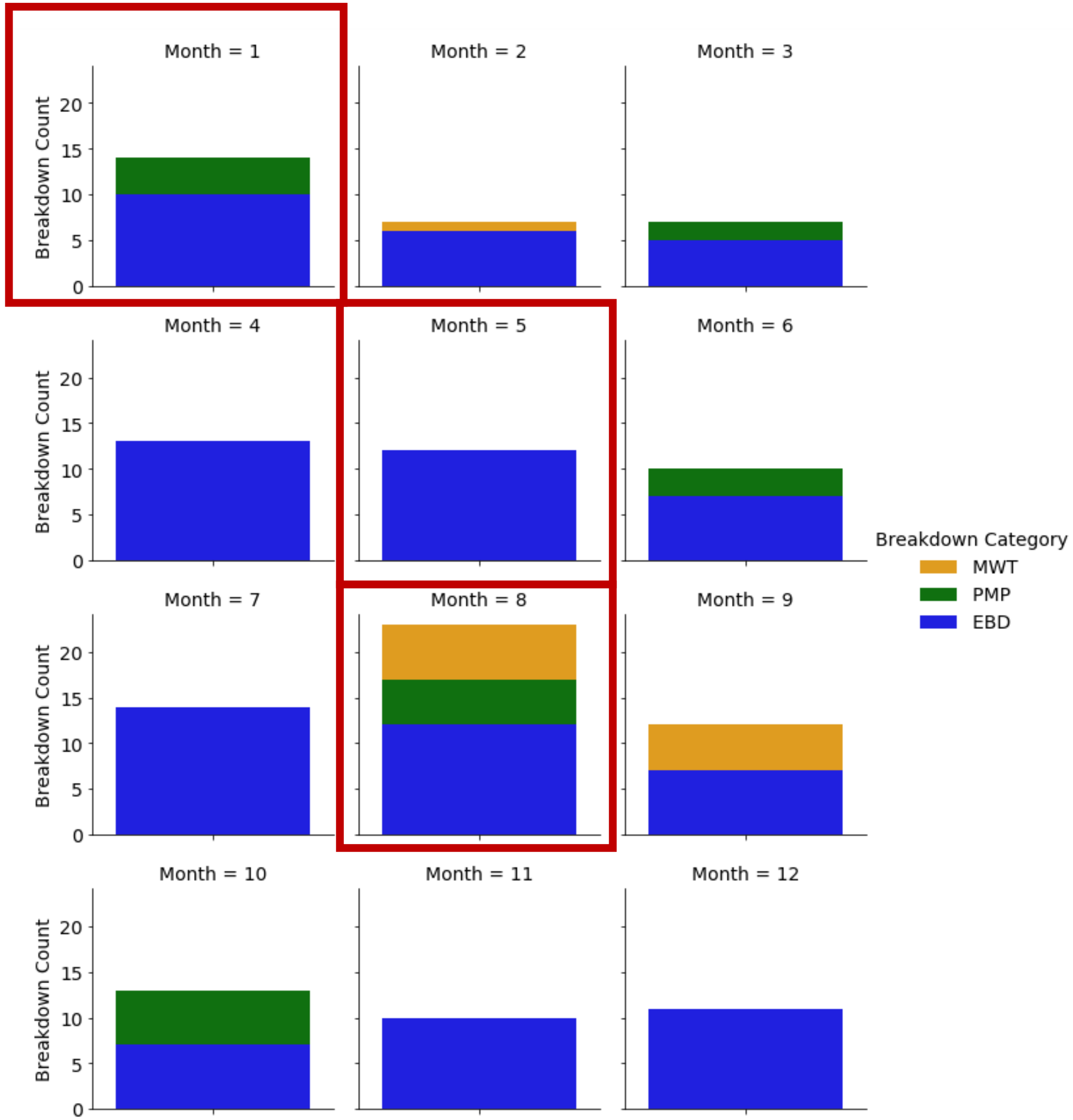


3-8 Breakdown Analysis

E. Top 3 Breakdown by Months

Critical Repair Months are:

- August (52 repairs)
- January (33 repairs)
- May (31 repairs)



CONFIDENTIAL

3-9 Breakdown Analysis

E. Top 3 Breakdown by Modules

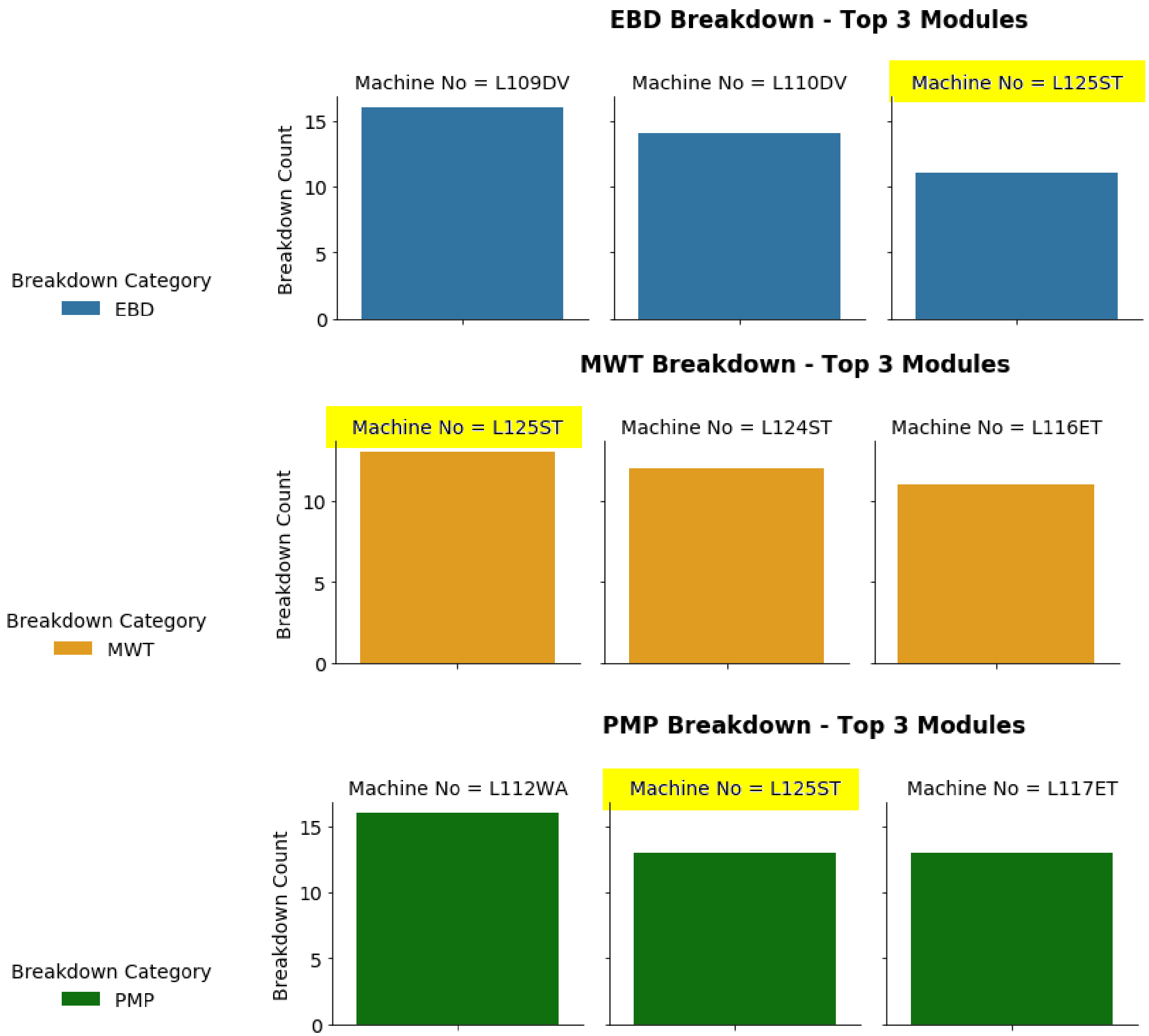
CONFIDENTIAL

3-10 Breakdown Analysis

E. Top 3 Modules in each Top 3 Breakdown

There is a common module that is in the top 3 modules of each top 3 breakdown:

- Machine No: L125ST
- Name: IL DES 4-Stripping 3
- Process: Stripping



CONFIDENTIAL

3-11 Breakdown Analysis Conclusion

Top 3 Breakdowns	Pattern/ Occurrence	Peak Month	Modules to Note
EDB	Increase Every 2 Months	July	L125ST
PMP	Peaks Every 2 Years	August	
MWT	Peaks During August	August	L112WA, L109DV, L110DV

CONFIDENTIAL

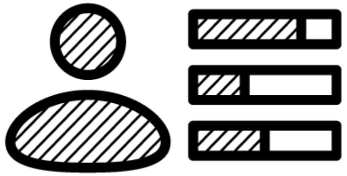
04 Repairer Analysis

Study of Repairers Overall Performance

4-1 Repairer Analysis

A. Criteria to Determine Top Repairer

Entries / Issues Resolved



Skills set (How many breakdown categories can repairer resolved)

Troubleshoot & Repair Time

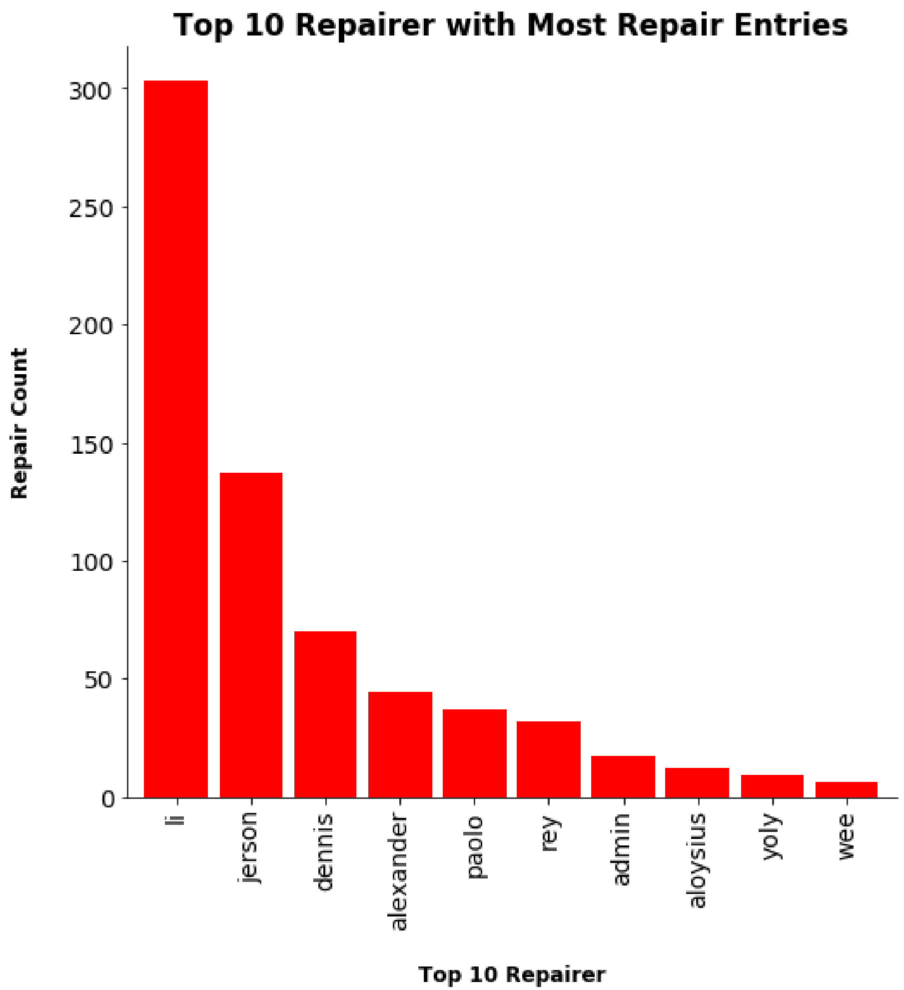
CONFIDENTIAL

4-2 Repairer Analysis

B. Top 10 Repairers Entries since Y2014 to Y2019

```
# Top 10 counts of entries/repairs done by each repairer
final_ca['Repairer Name'].value_counts().to_frame().head(10)
```

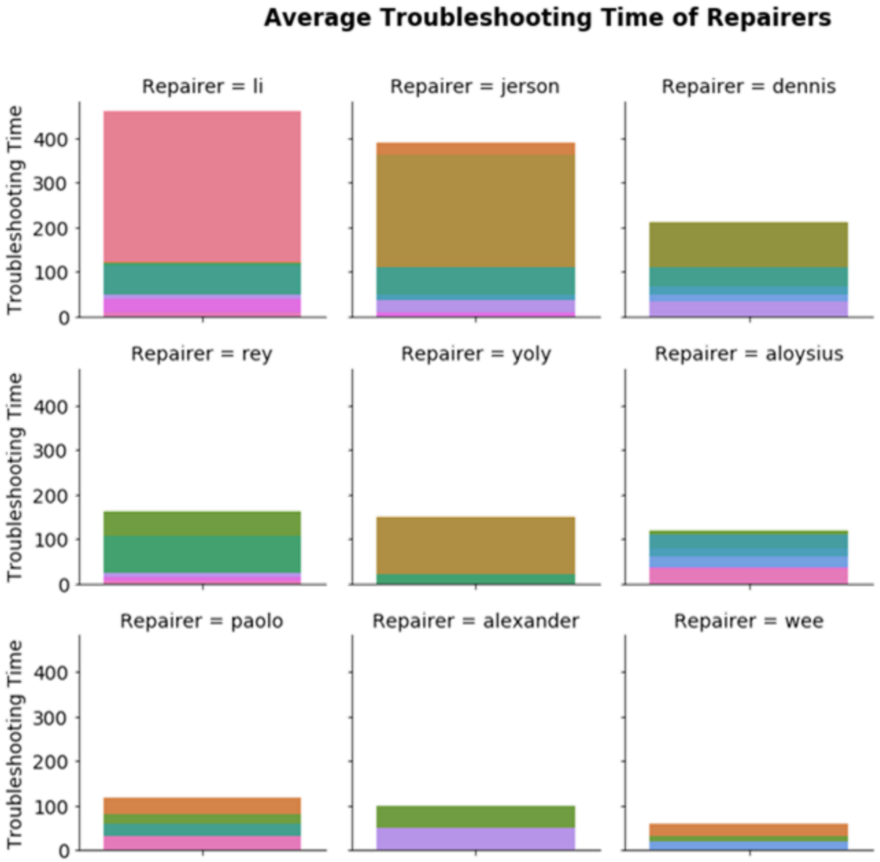
	Repairer Name
li	303
jerson	137
dennis	70
alexander	44
paolo	37
rey	32
admin	17
aloysius	12
yoly	9
wee	6

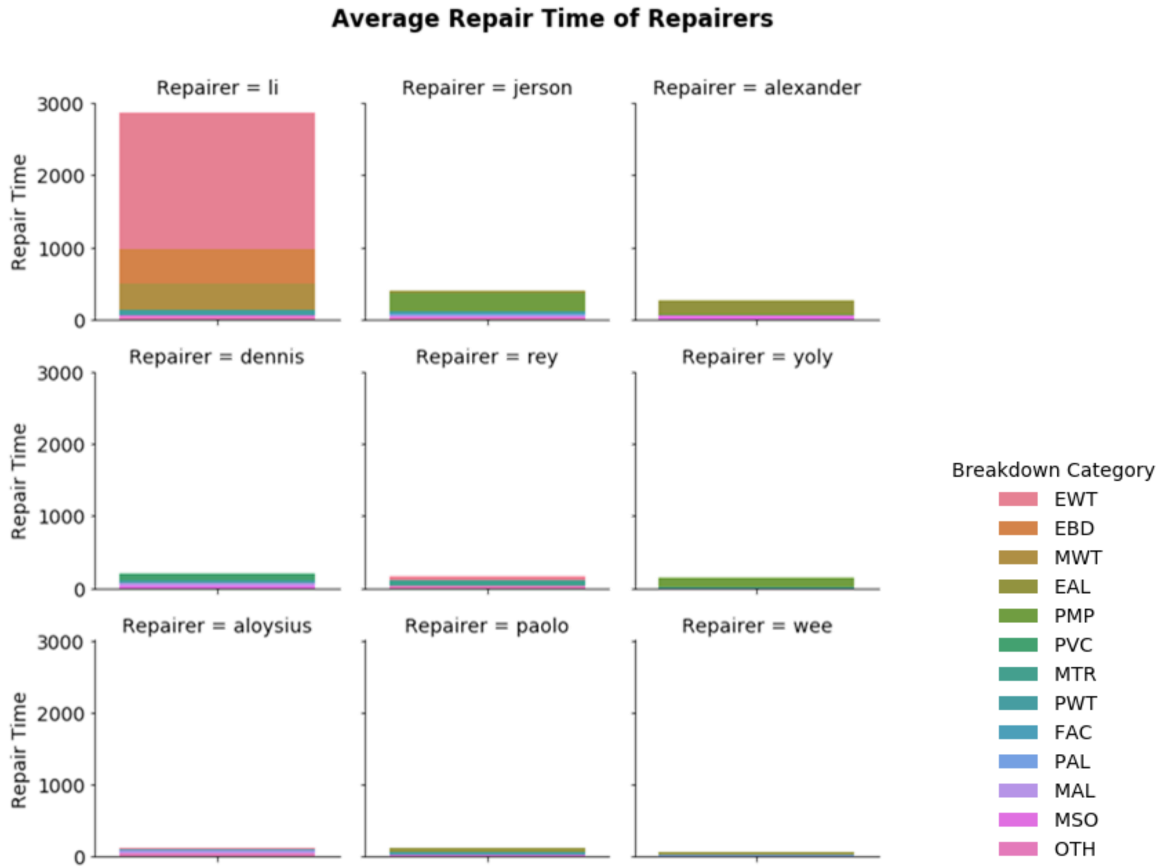


CONFIDENTIAL

4-3 Repairer Analysis

C. Overview of Top 10 Repairers Performance



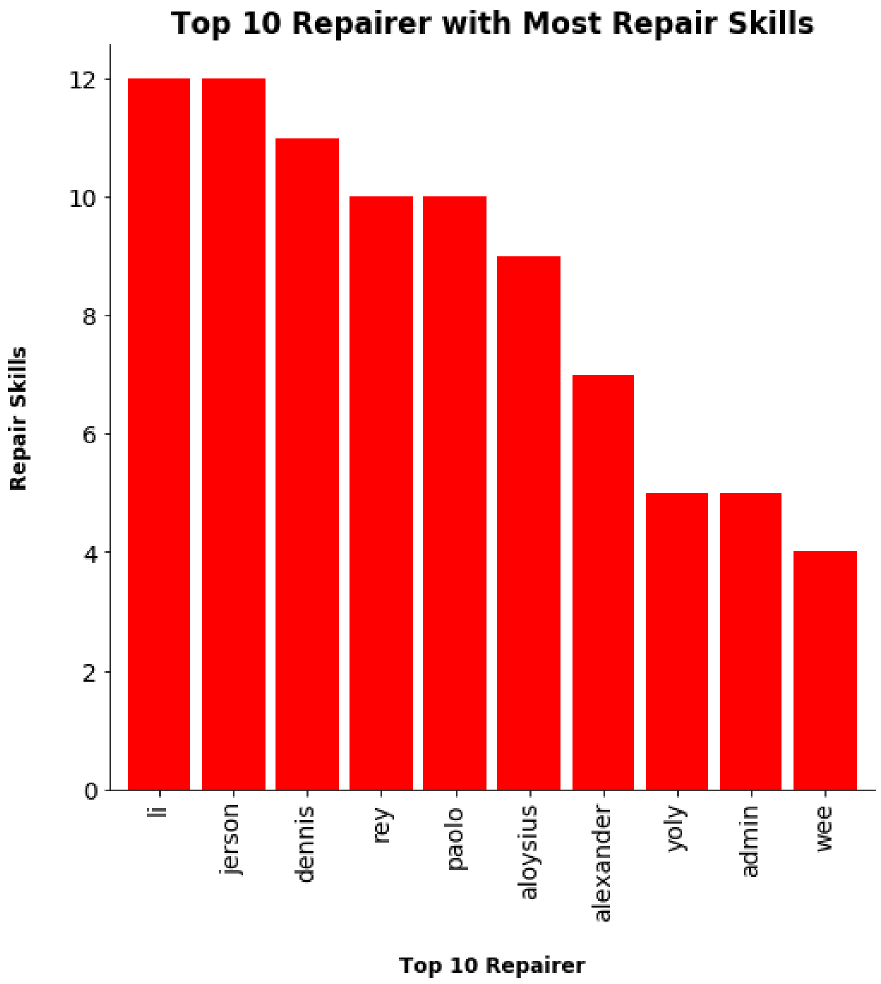


CONFIDENTIAL

4-4 Repairer Analysis

D. Top 10 Repairers with Most Repair Skills since Y2014 to Y2019

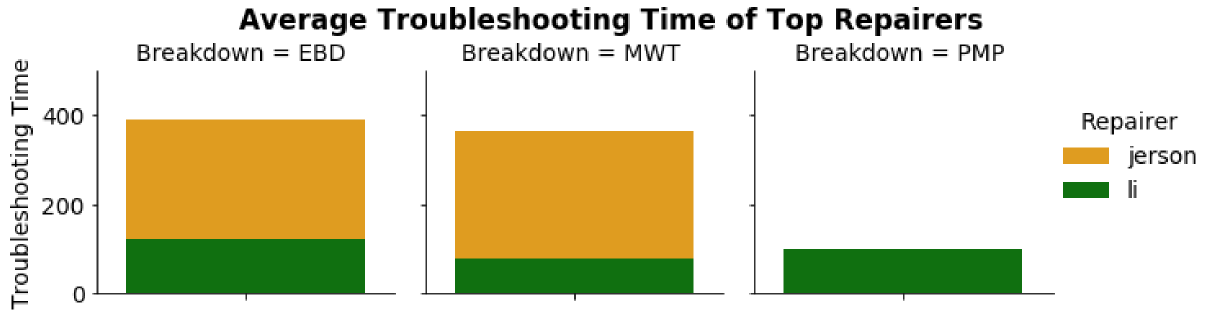
Repairer	Skills
li	12
jerson	12
dennis	11
rey	10
paolo	10
aloysius	9
alexander	7
yoly	5
admin	5
wee	4



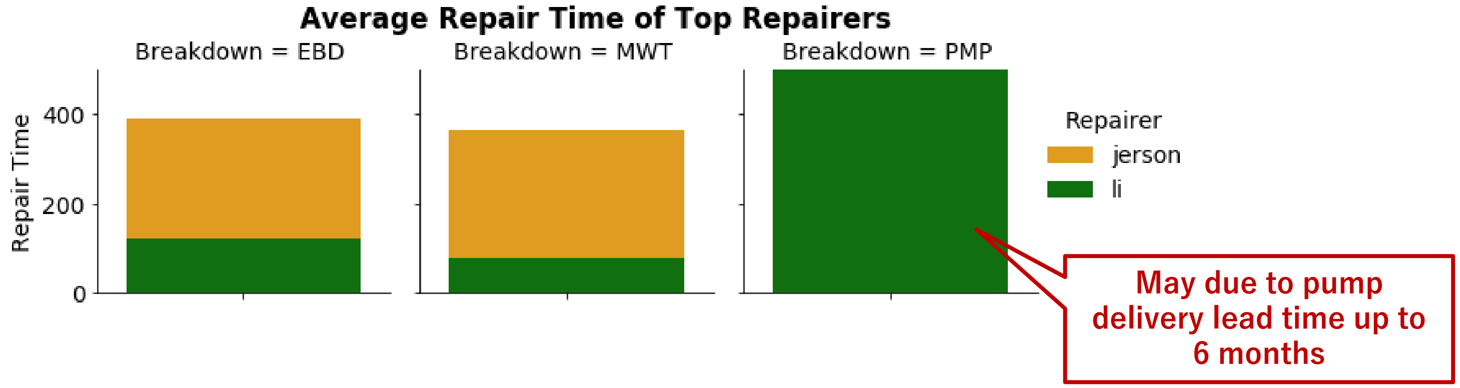
CONFIDENTIAL

4-5 Repairer Analysis

E. Closing on Top 2 Repairers



		Mean Troubleshooting Time
Breakdown	**Repairer**	
EBD	jerson	390.200000
	li	121.636364
MWT	jerson	362.777778
	li	77.177419
PMP	jerson	67.380952
	li	101.489362



		Mean Repair Time
Breakdown	**Repairer**	
EBD	jerson	390.600000
	li	122.181818
MWT	jerson	362.777778
	li	77.419355
PMP	jerson	67.857143
	li	2861.702128

3-11 Repairer Analysis Conclusion

Top 3 Repairer	No. of Repair Entries	No. of Skills	Troubleshoot Time (Mean)	Repair Time (Mean)
Li	**303**	**12**	**100.10 mins**	**1020.43 mins**
Jerson	137	12	273.45 mins	273.74 mins
Dennis	70	11	-	-

- Li did better overall when we compared troubleshoot and repair time against the top 3 breakdowns

05 Machine Learning

Predicting Troubleshoot & Repair Time

5-1 Predicting Troubleshoot & Repair Time

A. Encoding the Breakdown Categories for Machine Learning

In [79]:
```
# Replace string and organize the values of the columns

breakdown_cat = [
    (model['Breakdown Repair Category '] == 'EAL '),
    (model['Breakdown Repair Category '] == 'EBD '),
    (model['Breakdown Repair Category '] == 'EWT '),
    (model['Breakdown Repair Category '] == 'FAC '),
    (model['Breakdown Repair Category '] == 'MAL '),
    (model['Breakdown Repair Category '] == 'MSO '),
    (model['Breakdown Repair Category '] == 'MTR '),
    (model['Breakdown Repair Category '] == 'MWT '),
    (model['Breakdown Repair Category '] == 'OTH '),
    (model['Breakdown Repair Category '] == 'PAL '),
    (model['Breakdown Repair Category '] == 'PMP '),
    (model['Breakdown Repair Category '] == 'PVC '),
    (model['Breakdown Repair Category '] == 'PWT ')
    ]

# create a list of the values we want to assign for each condition
breakdown_values = ['0', '1', '2', '3', '4', '5', '6', '7', '8', '9', '10', '11', '12']

# create a new column and use np.select to assign values to it using our lists as arguments
model['BreakdownCat'] = np.select(breakdown_cat, breakdown_values)


# display updated DataFrame

model.head()
```

In [80]:
```
from sklearn.preprocessing import LabelEncoder

le = LabelEncoder()
model['BreakdownCat'] = model.apply(le.fit_transform)
model.head()
```

Out[80]:

	Breakdown Repair Category	Troubleshoot Hours (mins)	Repair Hours (mins)	Repairer Name	BreakdownCat
0	MWT	35.0	35.0	jerson	7
1	EBD	50.0	50.0	li	1
2	PVC	30.0	30.0	jerson	11
3	PWT	15.0	15.0	dennis	12
4	PWT	20.0	20.0	li	12

CONFIDENTIAL

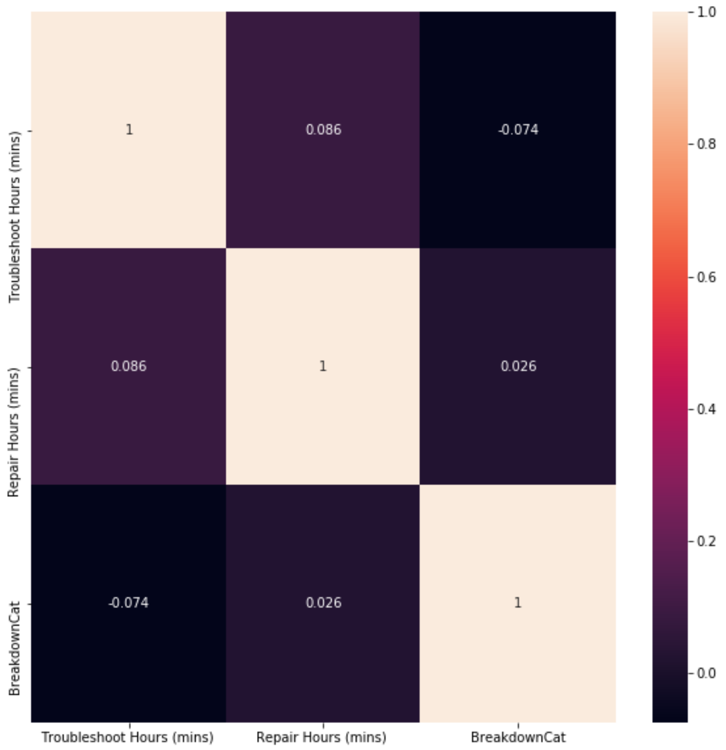
5-2 Predicting Troubleshoot & Repair Time

` B. Checking the Correlation of Features

```
In [81]: model.corr()
```

Out[81]:

	Troubleshoot Hours (mins)	Repair Hours (mins)	BreakdownCat
Troubleshoot Hours (mins)	1.000000	0.085649	-0.073771
Repair Hours (mins)	0.085649	1.000000	0.026329
BreakdownCat	-0.073771	0.026329	1.000000



- The features do not really have a correlation with each other

CONFIDENTIAL

5-3 Predicting Troubleshoot & Repair Time

C. Simple Model Using Linear Regression (Troubleshoot Time Prediction)

Troubleshoot Prediction (Linear Regression)

In [103]:
```
t_target = np.array(model['Troubleshoot Hours (mins) '])
```

In [104]:
```
t_features = np.array(model.drop(['Troubleshoot Hours (mins) ', 'Repair Hours (mins) ', 'Repairer Name', 'Troubleshoot Hours (mir
```

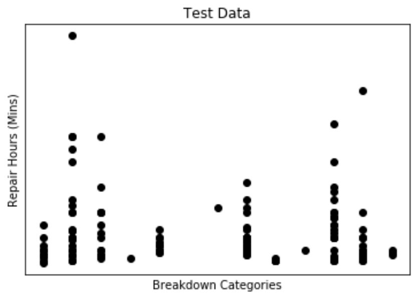
In [105]:
```
t_target = t_target.reshape(694,1)
r_features = t_features.reshape(694,1)
```

In [106]:
```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(t_features, t_target, test_size = 0.2, random_state = 123)
```

In [107]:
```
# Plot outputs
plt.scatter(X_test, y_test,  color='black')
plt.title('Test Data')
plt.xlabel('Breakdown Categories')
plt.ylabel('Repair Hours (Mins)')
plt.xticks(())
plt.yticks(())
```

Out[107]: ([], <a list of 0 Text yticklabel objects>)

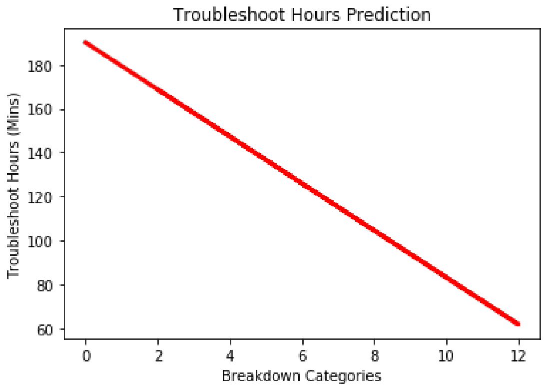


In [108]:
```
# Create linear regression object
t_regr = LinearRegression()

# Train the model using the training sets
t_regr.fit(X_train, y_train)

#regr.predict(X_test).reshape(1, 1))

# Plot outputs
plt.plot(X_test, t_regr.predict(X_test), color='red',linewidth=3)
plt.title('Troubleshoot Hours Prediction')
plt.xlabel('Breakdown Categories')
plt.ylabel('Troubleshoot Hours (Mins)')
plt.show()
```



5-4 Predicting Troubleshoot & Repair Time

D. Simple Model Using Linear Regression (Repair Time Prediction)

Repair Prediction (Linear Regression)

In [109]:
```
r_target = np.array(model['Repair Hours (mins) '])
```

In [110]:
```
r_features = np.array(model.drop(['Repair Hours (mins) ', 'Repairer Name', 'Troubleshoot Hours (mins) ', 'Breakdown Repair Categ
```

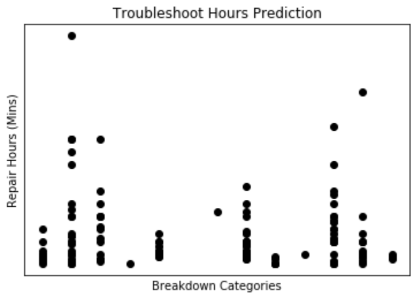
In [111]:
```
r_target = r_target.reshape(694,1)
r_features = r_features.reshape(694,1)
```

In [112]:
```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(r_features, r_target, test_size = 0.2, random_state = 123)
```

In [113]:
```
# Plot outputs
plt.scatter(X_test, y_test,  color='black')
plt.title('Troubleshoot Hours Prediction')
plt.xlabel('Breakdown Categories')
plt.ylabel('Repair Hours (Mins)')
plt.xticks(())
plt.yticks(())
```

Out[113]: ([], <a list of 0 Text yticklabel objects>)

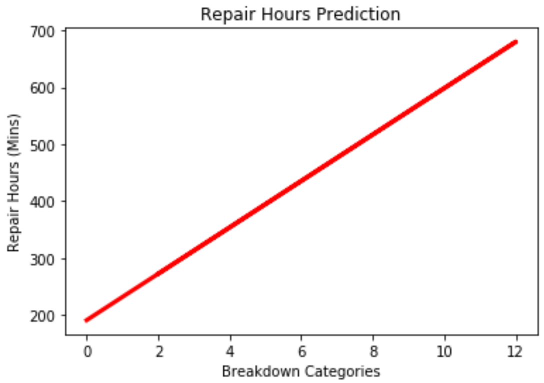


In [114]:
```
# Create linear regression object
r_regr = LinearRegression()

# Train the model using the training sets
r_regr.fit(X_train, y_train)

#regr.predict(X_test).reshape(1, 1))

# Plot outputs
plt.plot(X_test, r_regr.predict(X_test), color='red',linewidth=3)
plt.title('Repair Hours Prediction')
plt.xlabel('Breakdown Categories')
plt.ylabel('Repair Hours (Mins)')
plt.show()
```



5-4 Predicting Troubleshoot & Repair Time

D. Simple Model Using Linear Regression (Repair Time Prediction)

Repair Prediction (Linear Regression)

In [109]:
```
r_target = np.array(model['Repair Hours (mins) '])
```

In [110]:
```
r_features = np.array(model.drop(['Repair Hours (mins) ', 'Repairer Name', 'Troubleshoot Hours (mins) ', 'Breakdown Repair Categ
```

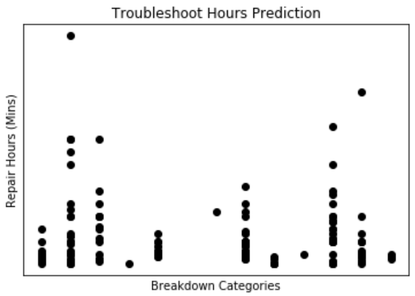
In [111]:
```
r_target = r_target.reshape(694,1)
r_features = r_features.reshape(694,1)
```

In [112]:
```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(r_features, r_target, test_size = 0.2, random_state = 123)
```

In [113]:
```
# Plot outputs
plt.scatter(X_test, y_test,  color='black')
plt.title('Troubleshoot Hours Prediction')
plt.xlabel('Breakdown Categories')
plt.ylabel('Repair Hours (Mins)')
plt.xticks(())
plt.yticks(())
```

Out[113]: ([], <a list of 0 Text yticklabel objects>)

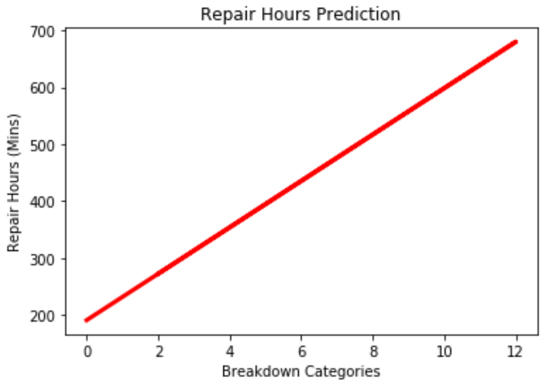


In [114]:
```
# Create linear regression object
r_regr = LinearRegression()

# Train the model using the training sets
r_regr.fit(X_train, y_train)

#regr.predict(X_test).reshape(1, 1))

# Plot outputs
plt.plot(X_test, r_regr.predict(X_test), color='red',linewidth=3)
plt.title('Repair Hours Prediction')
plt.xlabel('Breakdown Categories')
plt.ylabel('Repair Hours (Mins)')
plt.show()
```



CONFIDENTIAL

5-5 Predicting Troubleshoot & Repair Time

E. Predict Troubleshoot and Repair Time based on Breakdown Input

Breakdown Categories:

- EAL
- EBD
- EWT
- FAC
- MAL
- MSO
- MTR
- MWT
- OTH
- PAL
- PMP
- PVC
- PWT

```
In [118]: breakdown_type = input("Enter Breakdown Category: ")

if breakdown_type == 'EAL':
    print('Predicted Troubleshoot Hours (Mins) for', breakdown_type,':', int(np.round(t_regr.predict([[0]])), ), 'mins' )
    print('Predicted Repair Hours (Mins) for', breakdown_type,':', int(np.round(r_regr.predict([[0]]))), 'mins' )
elif breakdown_type == 'EBD':
    print('Predicted Troubleshoot Hours (Mins) for', breakdown_type,':', int(np.round(t_regr.predict([[1]]))), 'mins' )
    print('Predicted Repair Hours (Mins) for', breakdown_type,':', int(np.round(r_regr.predict([[1]]))), 'mins' )
elif breakdown_type == 'EWT':
    print('Predicted Troubleshoot Hours (Mins) for', breakdown_type,':', int(np.round(t_regr.predict([[2]]))), 'mins' )
    print('Predicted Repair Hours (Mins) for', breakdown_type,':', int(np.round(r_regr.predict([[2]]))), 'mins' )
elif breakdown_type == 'FAC':
    print('Predicted Troubleshoot Hours (Mins) for', breakdown_type,':', int(np.round(t_regr.predict([[3]]))), 'mins' )
    print('Predicted Repair Hours (Mins) for', breakdown_type,':', int(np.round(r_regr.predict([[3]]))), 'mins' )
elif breakdown_type == 'MAL':
    print('Predicted Troubleshoot Hours (Mins) for', breakdown_type,':', int(np.round(t_regr.predict([[4]]))), 'mins' )
    print('Predicted Repair Hours (Mins) for', breakdown_type,':', int(np.round(r_regr.predict([[4]]))), 'mins' )
elif breakdown_type == 'MSO':
    print('Predicted Troubleshoot Hours (Mins) for', breakdown_type,':', int(np.round(t_regr.predict([[5]]))), 'mins' )
    print('Predicted Repair Hours (Mins) for', breakdown_type,':', int(np.round(r_regr.predict([[5]]))), 'mins' )
elif breakdown_type == 'MTR':
    print('Predicted Troubleshoot Hours (Mins) for', breakdown_type,':', int(np.round(t_regr.predict([[6]]))), 'mins' )
    print('Predicted Repair Hours (Mins) for', breakdown_type,':', int(np.round(r_regr.predict([[6]]))), 'mins' )
elif breakdown_type == 'MWT':
    print('Predicted Troubleshoot Hours (Mins) for', breakdown_type,':', int(np.round(t_regr.predict([[7]]))), 'mins' )
    print('Predicted Repair Hours (Mins) for', breakdown_type,':', int(np.round(r_regr.predict([[7]]))), 'mins' )
elif breakdown_type == 'OTH':
    print('Predicted Troubleshoot Hours (Mins) for', breakdown_type,':', int(np.round(t_regr.predict([[8]]))), 'mins' )
    print('Predicted Repair Hours (Mins) for', breakdown_type,':', int(np.round(r_regr.predict([[8]]))), 'mins' )
elif breakdown_type == 'PAL':
    print('Predicted Troubleshoot Hours (Mins) for', breakdown_type,':', int(np.round(t_regr.predict([[9]]))), 'mins' )
    print('Predicted Repair Hours (Mins) for', breakdown_type,':', int(np.round(r_regr.predict([[9]]))), 'mins' )
elif breakdown_type == 'PMP':
    print('Predicted Troubleshoot Hours (Mins) for', breakdown_type,':', int(np.round(t_regr.predict([[10]]))), 'mins' )
    print('Predicted Repair Hours (Mins) for', breakdown_type,':', int(np.round(r_regr.predict([[10]]))), 'mins' )
elif breakdown_type == 'PVC':
    print('Predicted Troubleshoot Hours (Mins) for', breakdown_type,':', int(np.round(t_regr.predict([[11]]))), 'mins' )
    print('Predicted Repair Hours (Mins) for', breakdown_type,':', int(np.round(r_regr.predict([[11]]))), 'mins' )
elif breakdown_type == 'PWT':
    print('Predicted Troubleshoot Hours (Mins) for', breakdown_type,':', int(np.round(t_regr.predict([[12]]))), 'mins' )
    print('Predicted Repair Hours (Mins) for', breakdown_type,':', int(np.round(r_regr.predict([[12]]))), 'mins' )
```

Input:

```
Enter Breakdown Category:
```

Output:

```
Enter Breakdown Category: PMP
Predicted Troubleshoot Hours (Mins) for PMP : 83 mins
Predicted Repair Hours (Mins) for PMP : 599 mins
```

5-6 Predicting Troubleshoot & Repair Time

E. Predict Troubleshoot and Repair Time based on Top 3 Breakdowns

Breakdown Categories:

- EAL
- EBD
- EWT
- FAC
- MAL
- MSO
- MTR
- MWT
- OTH
- PAL
- PMP
- PVC
- PWT

Input:

```
Enter Breakdown Category:
```

Output:

```
Enter Breakdown Category: EBD
Predicted Troubleshoot Hours (Mins) for EBD : 179 mins
Predicted Repair Hours (Mins) for EBD : 231 mins
```

Output:

```
Enter Breakdown Category: PMP
Predicted Troubleshoot Hours (Mins) for PMP : 83 mins
Predicted Repair Hours (Mins) for PMP : 599 mins
```

Output:

```
Enter Breakdown Category: MWT
Predicted Troubleshoot Hours (Mins) for MWT : 115 mins
Predicted Repair Hours (Mins) for MWT : 476 mins
```

06 Final Conclusion

Insights, Recommendations and Conclusion

6-1 Conclusion/Recommendations

- **Machine (Objective #1)**

The company wants to be aware of the **top causes of downtime** and its **occurrence frequency/patterns** so as to plan ahead before the downtime occurs.

Top 3 Breakdowns	Pattern/ Occurrence	Peak Month	Modules to Note
EDB	Increase Every 2 Months	July	**L125ST** L112WA, L109DV, L110DV
PMP	Peaks Every 2 Years	August	
MWT	Peaks During August	August	

CONFIDENTIAL

6-2 Conclusion/Recommendations

- **Man (Objective #2)**

With efforts to recognize employees' performance, the company wants to reward the **best technician/repairs** over the years but do not know how to measure/judge their performance as they are considered a more technical role.

Top 3 Repairer	No. of Repair Entries	No. of Skills	Troubleshoot Time (Mean)	Repair Time (Mean)
Li	**303**	**12**	**100.10 mins**	**1020.43 mins**
Jerson	137	12	273.45 mins	273.74 mins
Dennis	70	11	-	-

- Li did better overall when we compared troubleshoot and repair time against the top 3 breakdowns

CONFIDENTIAL

6-3 Conclusion/Recommendations

- **Prediction (Additional)**

In order to reduce the overall downtime, the machine learning prediction enables a setting of target **Repair Hours** for the technician/repairer so as to look into those with long repair hours so to increase the Overall Equipment Efficiency (OEE).

Output:

```
Enter Breakdown Category: EBD
Predicted Troubleshoot Hours (Mins) for EBD : 179 mins
Predicted Repair Hours (Mins) for EBD : 231 mins
```

Output:

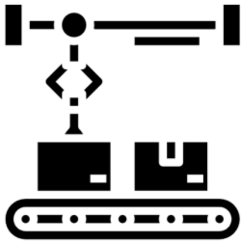
```
Enter Breakdown Category: PMP
Predicted Troubleshoot Hours (Mins) for PMP : 83 mins
Predicted Repair Hours (Mins) for PMP : 599 mins
```

Output:

```
Enter Breakdown Category: MWT
Predicted Troubleshoot Hours (Mins) for MWT : 115 mins
Predicted Repair Hours (Mins) for MWT : 476 mins
```

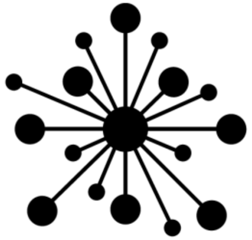
6-4 Conclusion/Recommendations

- What we aim to achieve after this:



Reduce Machine Breakdown

Improve Repairer’s Efficiency



Increase in Data Driven Decision Making

Thank You .

End of Presentation