Predicting Deferments

Jared Turner, Steven Sun, Ryan Leveille,
Jaime Flores, Andy Barnes

Scope


Data:
Candy Cane Well
Time Series
Unbalanced
Goals:
Predict deferment 24 hours in advance
Identify human shutdowns vs. system failures
Anomaly detection algorithm
KPIS:
Confusion Matrix
Validation Test
Visualization

Preprocessing

- Import data
- Explore data
- Check for NA's
- Rename columns
- Drop columns
- Convert data
- Replace negatives
- Statistical analysis
- Visualize data
- Correlation matrix

```
24 import pandas as pd
25
26 ### read in excel
27 og = pd.read_excel('og.xlsx')
28
29 ### rename columns w/o spaces
30 og = og.rename(columns = {"Wellhead Tubing - Pressure": "WellheadTubingPressure"})
31 og = og.rename(columns = {"Volume - Calendar Day Production": "Volume"})
32 og = og.rename(columns = {'Wellhead Casing "B" - Pressure': "CasingBPressure"})
33 og = og.rename(columns = {'Wellhead Casing "A" - Pressure': "CasingAPressure"})
34 og = og.rename(columns = {'Flowline Pressure': "FlowlinePressure"})
35 og = og.rename(columns = {'Flowline Temperature': "FlowlineTemperature"})
36 og = og.rename(columns = {'Name of Facility': "Name"})
37 og = og.rename(columns = {'Type of Facility': "Type"})
```

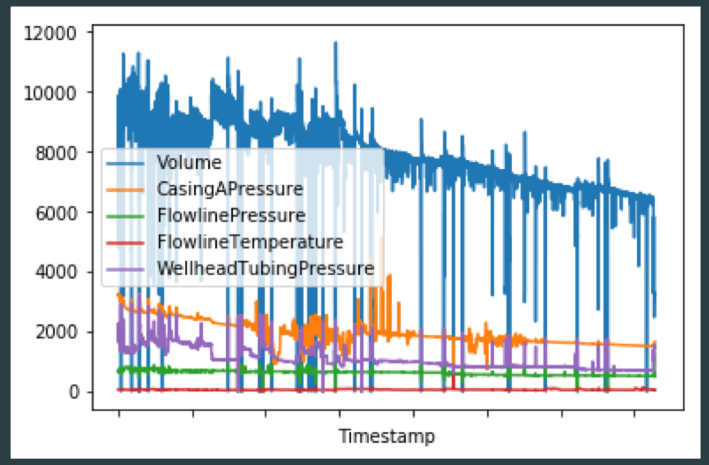
```
numna = og.isnull().sum()
##delete dataframes that are unneccessary - those include Nameo
ogclean = og.drop(columns = ['Name','Type', 'CasingBPressure'])
```

```
#change everything to float for calculations
ogclean['WellheadTubingPressure'] = ogclean.WellheadTubingPressure.astype(float)
ogclean['Volume'] = ogclean.Volume.astype(float)
ogclean['CasingAPressure'] = ogclean.CasingAPressure.astype(float)
ogclean['FlowlinePressure'] = ogclean.FlowlinePressure.astype(float)
ogclean['FlowlineTemperature'] = ogclean.FlowlineTemperature.astype(float)

#get rid of negatives in flowlinepressure
ogclean.loc[ogclean['FlowlinePressure']<0] = 0
```

Observation

This python visualization sucks
Steven perhaps add some badass R stuff here ☺



Categorization

3 Categories: (DEF,REG,HUM)

- Volume[i] >= 4000 = REG
- Volume[i] < 4000 = HUM
- Volume[i] != 0 in section. section = DEF

Sections: 136 (DEF,REG,HUM)

3 to 2 Categories: (DEF or NOT)

- If HUM or REG == NOT

Sections: 52 (DEF, NOT)

```
cats = pd.DataFrame(vol_list, columns=['Values']) #create new dataframe
cats['Categories'] = '' #create a new column in Cats that will consist
cats.loc[cats.Values>=4000, 'Categories'] = 'REG' #initial category (if
cats.loc[cats.Values<4000, 'Categories'] = 'HUM' #anything below 4000 we

cats['section'] = (cats.Categories != cats.Categories.shift()).cumsum()
```

```
for n, g in cats.groupby('section'): #search through section by gr
    if 0 not in g.Values.values and 'HUM' in g.Categories.values:
        cats.loc[g.index, 'Categories'] = 'DEF' #this locates all
```

```
for n, g in finalcats.groupby('section'):
    if 'HUM' in g.Categories.values:
        finalcats.loc[g.index, 'Categories'] = 'NOT'

for n, g in finalcats.groupby('section'):
    if 'REG' in g.Categories.values:
        finalcats.loc[g.index, 'Categories'] = 'NOT'
```

```
sectionsize = finalcats.groupby(['section']).size()
```

section	
1	3344
2	14
3	5656
4	44
5	13171
6	44
7	2190
8	15
9	3285
10	270
11	15
12	45
13	1726
14	14
15	4995
16	15
17	4515
18	165
19	17056
20	14

Histogram/Probability/Logistic Regression/Classification Tree

Nuerals

Clustering

Visualization

KPI