# PDD EDA

August 29, 2021

### 0.0.1 Data exploring

Let's look at the data without any preprocesses steps, if we are lucky enough, we may find something interesting

[75]:
```
!pip install pandas
!pip install seaborn
!pip install matplotlib
!pip install wordcloud
!pip install nltk
```

```
Requirement already satisfied: pandas in
/home/yaroslav/miniconda3/lib/python3.7/site-packages (1.1.3)
Requirement already satisfied: python-dateutil>=2.7.3 in
/home/yaroslav/miniconda3/lib/python3.7/site-packages (from pandas) (2.8.1)
Requirement already satisfied: pytz>=2017.2 in
/home/yaroslav/miniconda3/lib/python3.7/site-packages (from pandas) (2020.5)
Requirement already satisfied: numpy>=1.15.4 in
/home/yaroslav/miniconda3/lib/python3.7/site-packages (from pandas) (1.19.5)
Requirement already satisfied: six>=1.5 in
/home/yaroslav/miniconda3/lib/python3.7/site-packages (from python-
dateutil>=2.7.3->pandas) (1.14.0)
Requirement already satisfied: seaborn in
/home/yaroslav/miniconda3/lib/python3.7/site-packages (0.11.1)
Requirement already satisfied: matplotlib>=2.2 in
/home/yaroslav/miniconda3/lib/python3.7/site-packages (from seaborn) (3.4.1)
Requirement already satisfied: pandas>=0.23 in
/home/yaroslav/miniconda3/lib/python3.7/site-packages (from seaborn) (1.1.3)
Requirement already satisfied: numpy>=1.15 in
/home/yaroslav/miniconda3/lib/python3.7/site-packages (from seaborn) (1.19.5)
Requirement already satisfied: scipy>=1.0 in
/home/yaroslav/miniconda3/lib/python3.7/site-packages (from seaborn) (1.6.1)
Requirement already satisfied: cycler>=0.10 in
/home/yaroslav/miniconda3/lib/python3.7/site-packages (from
matplotlib>=2.2->seaborn) (0.10.0)
Requirement already satisfied: pyparsing>=2.2.1 in
/home/yaroslav/miniconda3/lib/python3.7/site-packages (from
matplotlib>=2.2->seaborn) (2.4.7)
Requirement already satisfied: python-dateutil>=2.7 in
```

```
/home/yaroslav/miniconda3/lib/python3.7/site-packages (from
matplotlib>=2.2->seaborn) (2.8.1)
Requirement already satisfied: kiwisolver>=1.0.1 in
/home/yaroslav/miniconda3/lib/python3.7/site-packages (from
matplotlib>=2.2->seaborn) (1.2.0)
Requirement already satisfied: pillow>=6.2.0 in
/home/yaroslav/miniconda3/lib/python3.7/site-packages (from
matplotlib>=2.2->seaborn) (7.2.0)
Requirement already satisfied: pytz>=2017.2 in
/home/yaroslav/miniconda3/lib/python3.7/site-packages (from
pandas>=0.23->seaborn) (2020.5)
Requirement already satisfied: six in
/home/yaroslav/miniconda3/lib/python3.7/site-packages (from
cycler>=0.10->matplotlib>=2.2->seaborn) (1.14.0)
Requirement already satisfied: matplotlib in
/home/yaroslav/miniconda3/lib/python3.7/site-packages (3.4.1)
Requirement already satisfied: pillow>=6.2.0 in
/home/yaroslav/miniconda3/lib/python3.7/site-packages (from matplotlib) (7.2.0)
Requirement already satisfied: numpy>=1.16 in
/home/yaroslav/miniconda3/lib/python3.7/site-packages (from matplotlib) (1.19.5)
Requirement already satisfied: python-dateutil>=2.7 in
/home/yaroslav/miniconda3/lib/python3.7/site-packages (from matplotlib) (2.8.1)
Requirement already satisfied: kiwisolver>=1.0.1 in
/home/yaroslav/miniconda3/lib/python3.7/site-packages (from matplotlib) (1.2.0)
Requirement already satisfied: pyparsing>=2.2.1 in
/home/yaroslav/miniconda3/lib/python3.7/site-packages (from matplotlib) (2.4.7)
Requirement already satisfied: cycler>=0.10 in
/home/yaroslav/miniconda3/lib/python3.7/site-packages (from matplotlib) (0.10.0)
Requirement already satisfied: six>=1.5 in
/home/yaroslav/miniconda3/lib/python3.7/site-packages (from python-
dateutil>=2.7->matplotlib) (1.14.0)
Collecting wordcloud
  Downloading wordcloud-1.8.1-cp37-cp37m-manylinux1_x86_64.whl (366 kB)
     |                 | 366 kB 1.2 MB/s eta 0:00:01
Requirement already satisfied: numpy>=1.6.1 in
/home/yaroslav/miniconda3/lib/python3.7/site-packages (from wordcloud) (1.19.5)
Requirement already satisfied: matplotlib in
/home/yaroslav/miniconda3/lib/python3.7/site-packages (from wordcloud) (3.4.1)
Requirement already satisfied: pillow in
/home/yaroslav/miniconda3/lib/python3.7/site-packages (from wordcloud) (7.2.0)
Requirement already satisfied: kiwisolver>=1.0.1 in
/home/yaroslav/miniconda3/lib/python3.7/site-packages (from
matplotlib->wordcloud) (1.2.0)
Requirement already satisfied: cycler>=0.10 in
/home/yaroslav/miniconda3/lib/python3.7/site-packages (from
matplotlib->wordcloud) (0.10.0)
Requirement already satisfied: pyparsing>=2.2.1 in
/home/yaroslav/miniconda3/lib/python3.7/site-packages (from
```

```
matplotlib->wordcloud) (2.4.7)
Requirement already satisfied: python-dateutil>=2.7 in
/home/yaroslav/miniconda3/lib/python3.7/site-packages (from
matplotlib->wordcloud) (2.8.1)
Requirement already satisfied: six in
/home/yaroslav/miniconda3/lib/python3.7/site-packages (from
cycler>=0.10->matplotlib->wordcloud) (1.14.0)
Installing collected packages: wordcloud
Successfully installed wordcloud-1.8.1
Requirement already satisfied: nltk in
/home/yaroslav/miniconda3/lib/python3.7/site-packages (3.6.2)
Requirement already satisfied: joblib in
/home/yaroslav/.local/lib/python3.7/site-packages (from nltk) (1.0.1)
Requirement already satisfied: click in
/home/yaroslav/miniconda3/lib/python3.7/site-packages (from nltk) (7.1.2)
Requirement already satisfied: tqdm in
/home/yaroslav/miniconda3/lib/python3.7/site-packages (from nltk) (4.60.0)
Requirement already satisfied: regex in
/home/yaroslav/miniconda3/lib/python3.7/site-packages (from nltk) (2021.4.4)
```

[29]:
```
import pandas as pd
import seaborn as sns, numpy as np
sns.set_theme(); np.random.seed(0)
import matplotlib.pyplot as plt
from wordcloud import WordCloud
import nltk
import urllib3.request
```

[2]:
```
df = pd.read_json("/home/yaroslav/Desktop/study/research/all-issues.json")
```

let's try select set of fields, that we think can help us and look at the report

### 0.0.2 First hypothesis

we have enough datapoints in labels field if so, the task can be solved as the classification one

[3]:
```
import json
with open('all-issues.json') as file:
    data = json.load(file)
```

[4]:
```
label_list = [issue['labels'] for issue in data['issues'] if 'labels' in issue]
label_dataset = []
for labels in label_list:
     label_dataset.append([label['name'] for label in labels if 'name' in␣
 ↪label])
df = pd.DataFrame(label_dataset)
```

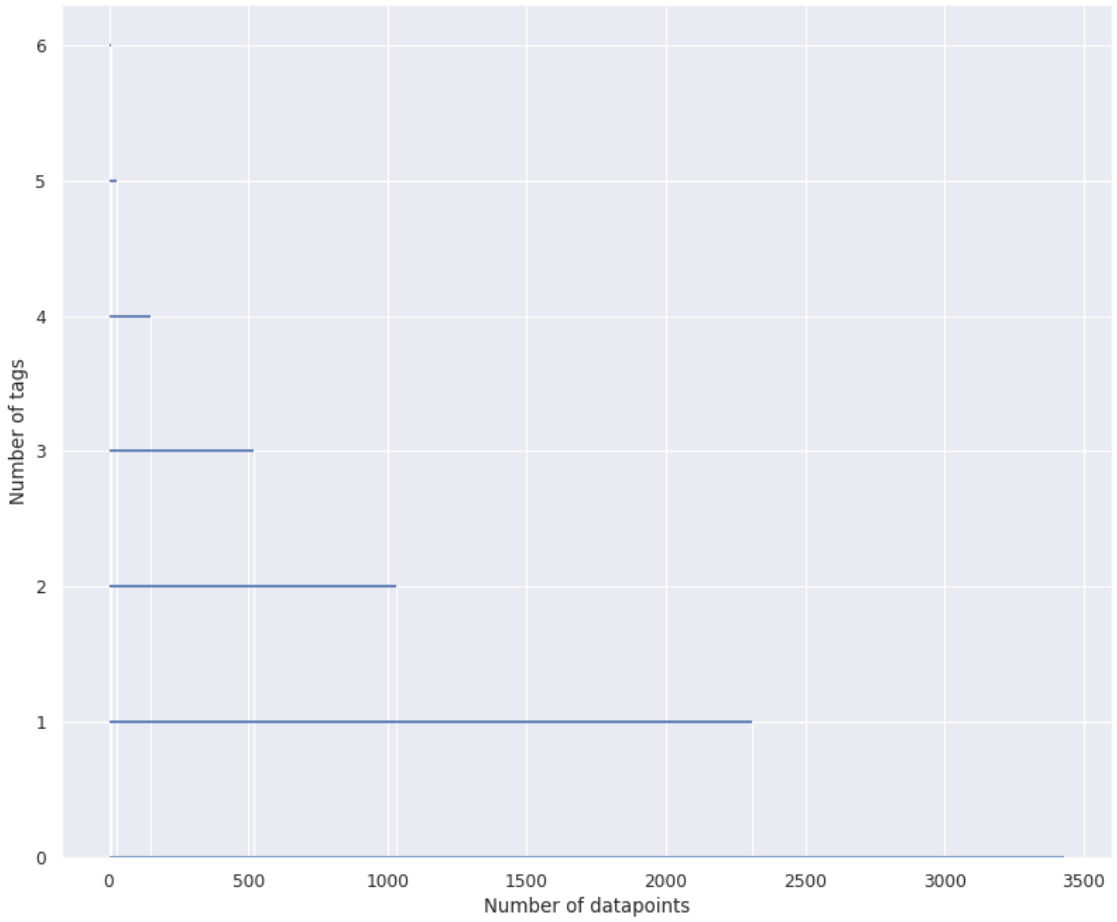
### 0.0.3 Results of the investigating of first hypothesis

[5]:
```
print('1. Number of datapoints:', df.shape[0])
print('2. Number of datapoints with at least one tag:', df.shape[0] -  df[0].
 ↪isnull().sum())
print('3. Number of datapoints with at least 2 tags', df.shape[0] - df[1].
 ↪isnull().sum())
```

1. Number of datapoints: 5661
2. Number of datapoints with at least one tag: 3429
3. Number of datapoints with at least 2 tags 2311

[6]:
```
x = [df.shape[0] - df[i].isnull().sum() for i in range(7)]

plt.figure(figsize=(12, 10), dpi=80)
plt.bar(x,[i for i in range(7)],align='center')
plt.xlabel('Number of datapoints')
plt.ylabel('Number of tags')
for i in range(7):
    plt.hlines(i,0,x[i])
plt.show()
```

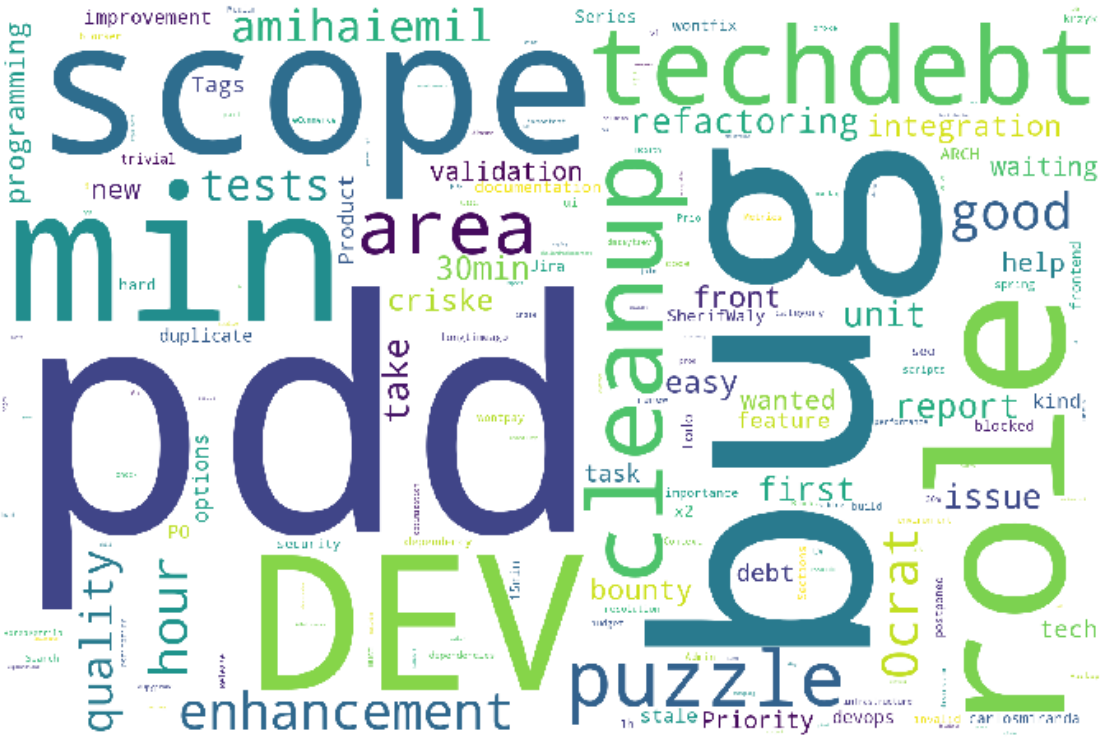


So we have pretty huge number of datapoints with at least one tag. But are they really helpful? I mean, is this label truly display severity of the puzzle?

[7]:
```
import itertools
```

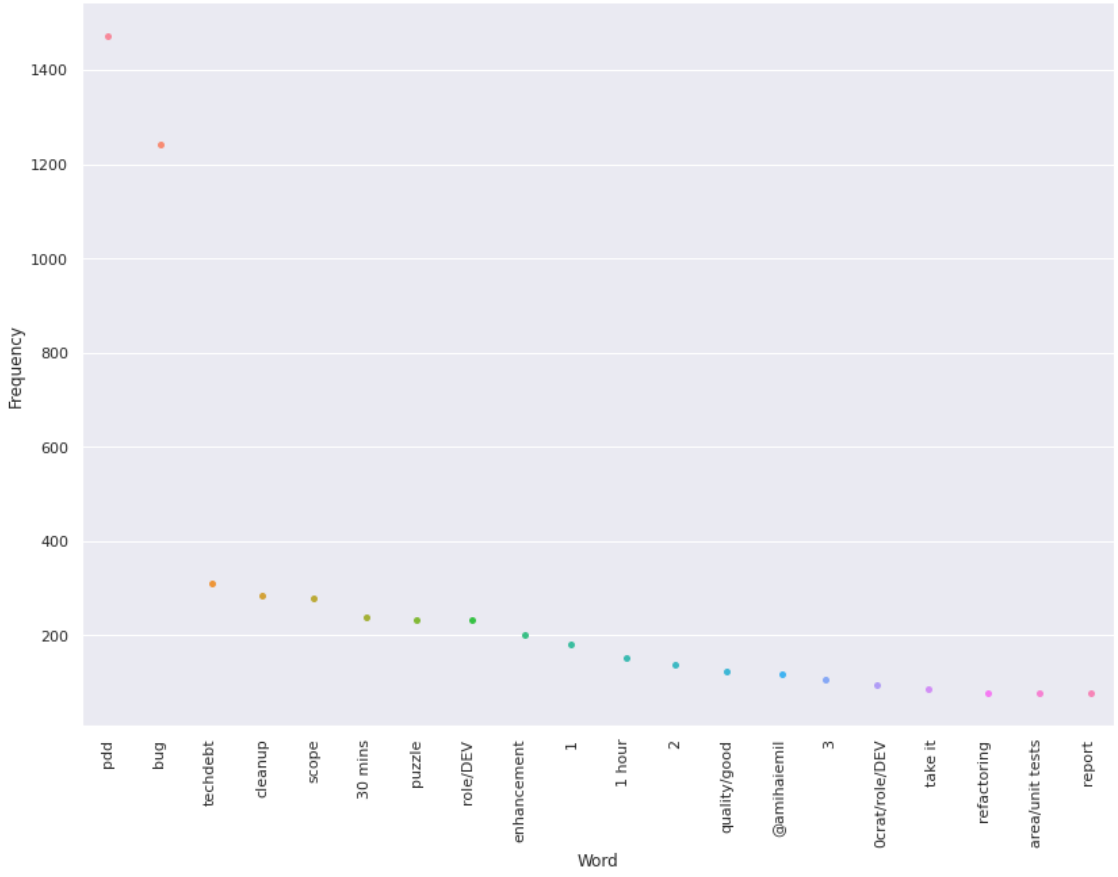
[8]:
```
tags = [tag for tag in list(itertools.chain(*[list(df[i]) for i in range(7)]))␣
↪if not pd.isnull(tag)]
tags_text = ' '.join(tags)
word_cloud = WordCloud(width=1800, height=1200,collocations = False,␣
↪background_color = 'white').generate(tags_text)
```

[9]:
```
plt.figure(figsize=(12, 10))
plt.imshow(word_cloud, interpolation='nearest')
plt.axis("off")
plt.show()
```




[10]:
```
word_dist = nltk.FreqDist(tags)
rslt=pd.DataFrame(word_dist.most_common(20),columns=['Word','Frequency'])
```

[11]:
```
sns.catplot(x='Word', y='Frequency',data = rslt,height=8.27, aspect=11.7/8.27);
plt.xticks(rotation=90);
```




#### 0.0.4 From the observed data i can make several conclusion:

1. There are some good labels of puzzles, that actually contains information about severity of the bug, or the importance for the client, like: hour, easy, enhancement, first, documentation, techdebt.
2. Most of the puzzles contains labels, which is not very relevant to the ranking task, for example can we understand the importance of task with label bug or pdd?

#### 0.0.5 As the conclusion for the first hypothesis, i propose to not to use the label field

#### 0.0.6 Second hypothesis: if task is hard, many puzzles will be created, so if we have a long series of related puzzles, we can think of it as a one feature that need to be implemented. Longer the puzzle chain, then more importantly will be head of the chain

[12]:
```
df = pd.read_csv('all-puzzles.csv')
df['Parent ID'].isnull().sum()
```

[12]: 10500

All puzzles do not have parent data, which means that we do not have data of relation of puzzles

### 0.0.7 At this moment, we can't use this approach

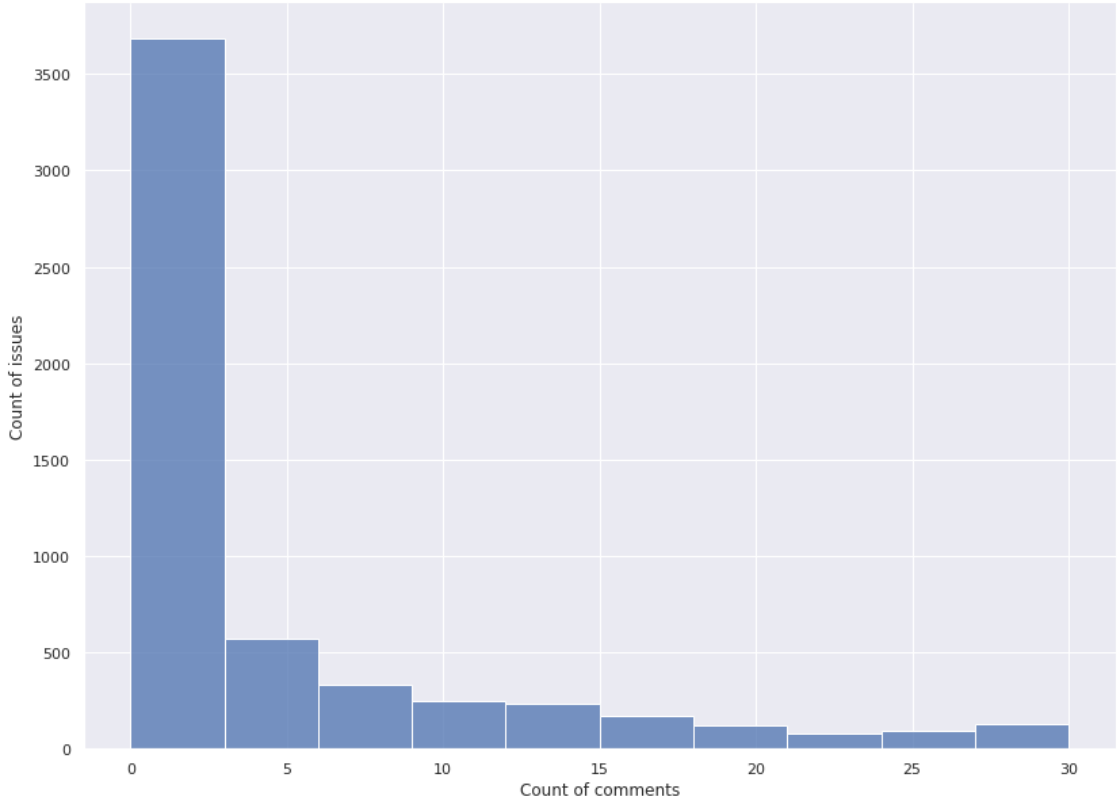
### 0.0.8 Another approach

If task is hard many people will be involved into this task, so we can predict the rating of the task, by predicting number of comments to the issue

[58]:
```
with open('all-issues2.json') as f:
    comments_count = []
    data = json.loads(f.read())
    data = [issue for issue in data['issues'] if 'comments_url' in issue]
    for issue in data:
        comments_count.append(len(issue['comments_url']))
```

[69]:
```
ax = sns.displot(comments_count,binwidth=3, height=8.27, aspect=11.7/8.27)
ax.set(xlabel='Count of comments', ylabel='Count of issues')
```

[69]: <seaborn.axisgrid.FacetGrid at 0x7fbd3d099910>




[74]:
```
print('Count of issues where count of messages at least one: ',␣
 ↪len(list(filter(lambda x: x >= 1,comments_count))))
print('Count of issues where count of messages is zero: ',␣
 ↪len(list(filter(lambda x: x == 0, comments_count))))
```

```
Count of issues where count of messages at least one:  4517
Count of issues where count of messages is zero:  1144
```

So even if we have very unbalanced data, where most of the issues have number of comments is less then 5, we could try this approach, when we predicting the rank of the puzzle, but predicting the count of comments