Nikhil Panse - IIT Guwahati

Importing text file and splitting it to create data variables

In [154]:

```
import pandas as pd
data = [line.strip() for line in open("train_set.txt", "r").readlines()]
data = [{"LineNumber": line.split("~")[0], "Number": line.split("~")[1].split("IN
data = pd.DataFrame(data)
dftest = [line.strip() for line in open("public_test_set.txt", "r").readlines()]
dftest = [{"LineNumber": line.split("~")[0], "Number": line.split("~")[1].split("
dftest = pd.DataFrame(dftest)
dp = [line.strip() for line in open("private_test_set.txt", "r").readlines()]
dp = [{"LineNumber": line.split("~")[0], "Number": line.split("~")[1].split("INFO
dp = pd.DataFrame(dp)
```

In [155]:

```
dftest
```

5	L5	"GET /v2/2019-09-10 meta_data.json HTTP/1.1" ...	62161	req-y5tj-ufwp-34nme6-6do12x	gs.api.openapi.compute.wsgi
6	L6	HTTP exception thrown: No entry found for any...	13691	req-x52i0-95ewzi-tnxuz	gs.openstack.wsgi.server
7	L7	"GET /v2/2019-09-12 meta_data.json HTTP/1.1" ...	25112	req-joh69-ydjr7v-rhezju-a7nq-rytxe8	gs.openapi.wsgi.server
8	L8	Instance spawned correctly	94733	req-vz2nx-5eva8-78jxq	gs.api.openapi.compute.wsgi

Importing data as text file and separating by new lines.

Using Term Frequency- Inverse Document Frequency to understand word frequencies

In [180]:

```
with open('train_set.txt') as f:
    file = [line.rstrip('\n') for line in f]

from sklearn.feature_extraction.text import TfidfVectorizer
vectorizer = TfidfVectorizer(analyzer="word", max_features=125)
X = vectorizer.fit_transform(file)
from scipy.sparse import csr_matrix
X = csr_matrix.todense(X)

```

K Means clustering

In [190]:

```
from sklearn.cluster import KMeans
kmeans = KMeans(n_clusters=18, random_state=0).fit(X)
```

Public dataset

In [191]:

```
with open('public_test_set.txt') as f:
    test = [line.rstrip('\n') for line in f]
vectorizer = TfidfVectorizer(analyzer="word", max_features=125)
Xt = vectorizer.fit_transform(test)
Xt = csr_matrix.todense(Xt)

```

Clustering public dataset

In [192]:

```
d = {"LineNumber":dftest["LineNumber"].values, "Class":kmeans.predict(Xt)}
df = pd.DataFrame(d)
df.to_csv("public_test.csv", index=False)

```

Private Dataset

In [193]:

```
with open('private_test_set.txt') as f:
    testp = [line.rstrip('\n') for line in f]
vectorizer = TfidfVectorizer(analyzer="word", max_features=125)
Xp = vectorizer.fit_transform(testp)
Xp = csr_matrix.todense(Xp)
p = {"LineNumber":dp["LineNumber"].values, "Class":kmeans.predict(Xp)}

dp = pd.DataFrame(p)
dp.to_csv("private_test.csv", index=False)
```

In [ ]:

```

```