# Analysis Task

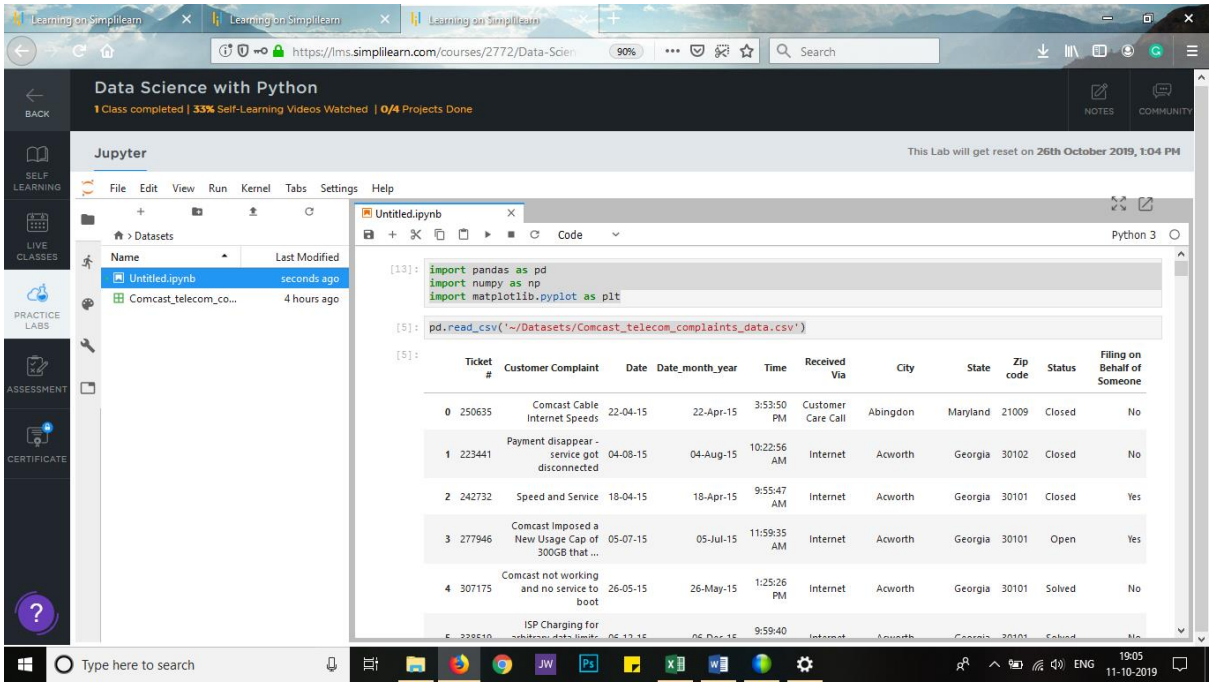
## Comcast Telecom Consumer Complaints Project

**To perform these tasks, you can use any of the different Python libraries such as NumPy, SciPy, Pandas, scikit-learn, matplotlib, and BeautifulSoup.**

- Import data into Python environment.

## Code

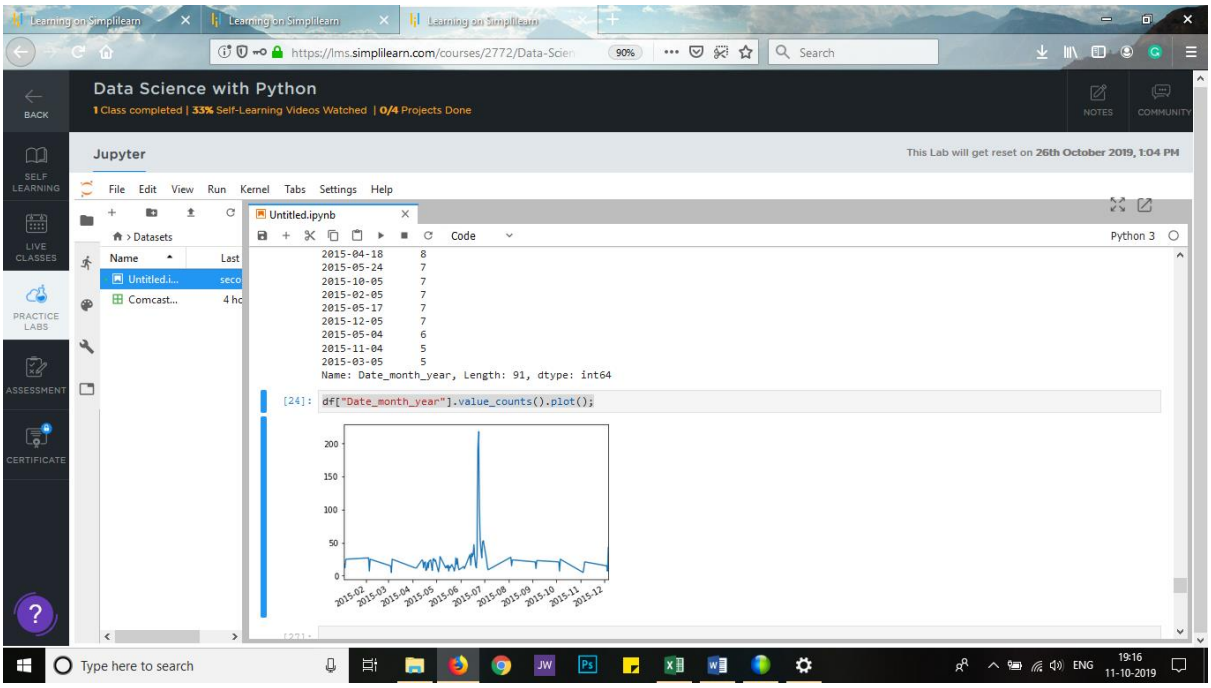
```
import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

pd.read_csv('~/Datasets/Comcast_telecom_complaints_data.csv')

df = pd.read_csv('~/Datasets/Comcast_telecom_complaints_data.csv')
```



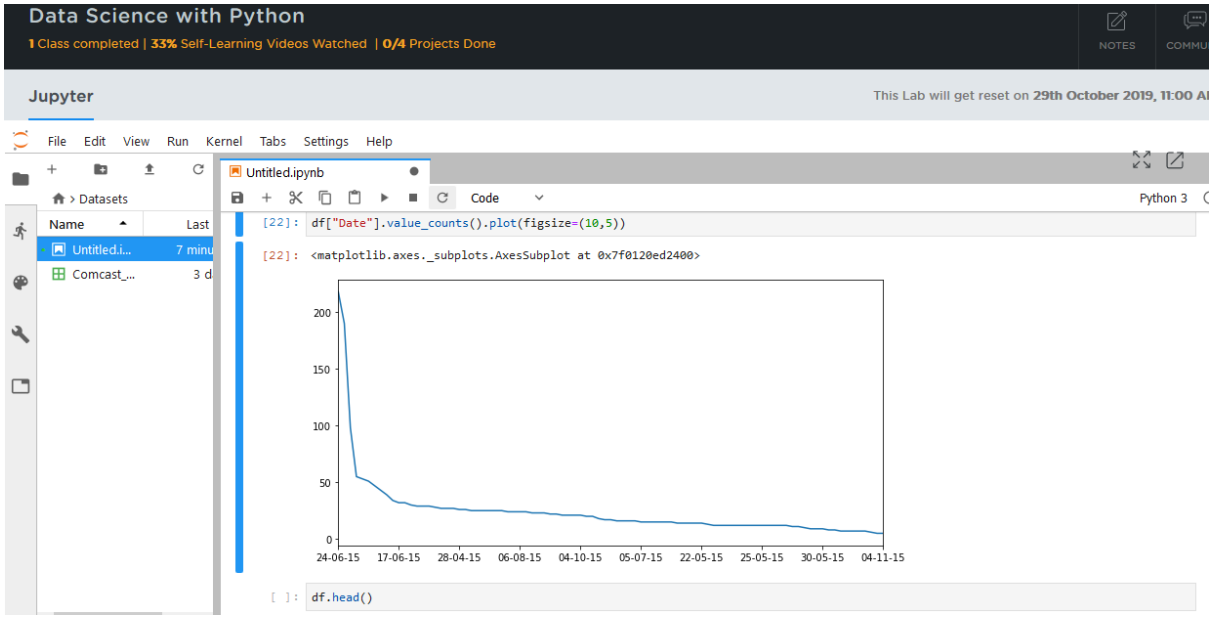
**Provide the trend chart for the number of complaints at monthly and daily granularity levels.**

## Code

df["Date_month_year"].value_counts().plot();



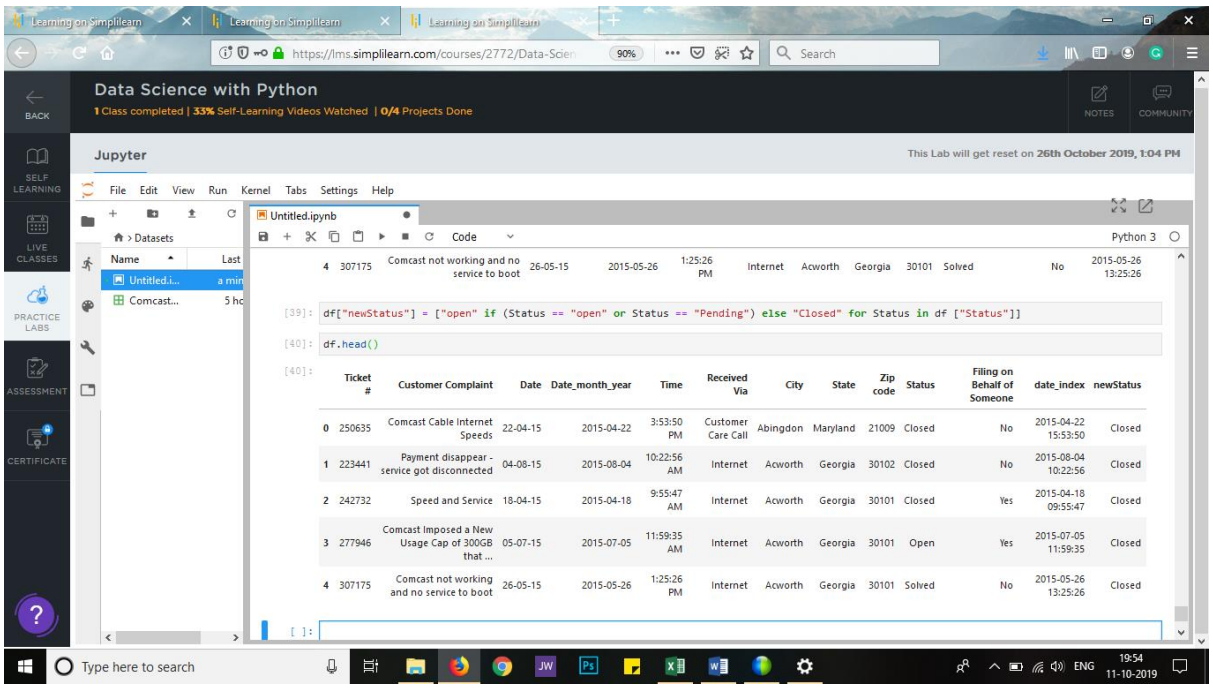

df["Date"].value_counts().plot(figsize=(10,5))



**Create a new categorical variable with value as** Open **and** Closed**. Open & Pending is to be categorized as Open and Closed & Solved is to be categorized as Closed.**

## Code

df["newStatus"] = ["open" if (Status == "open" or Status == "Pending") else "Closed" for Status in df ["Status"]]




**Provide state wise status of complaints in a stacked bar chart. Use the categorized variable from Q3. Provide insights on:**

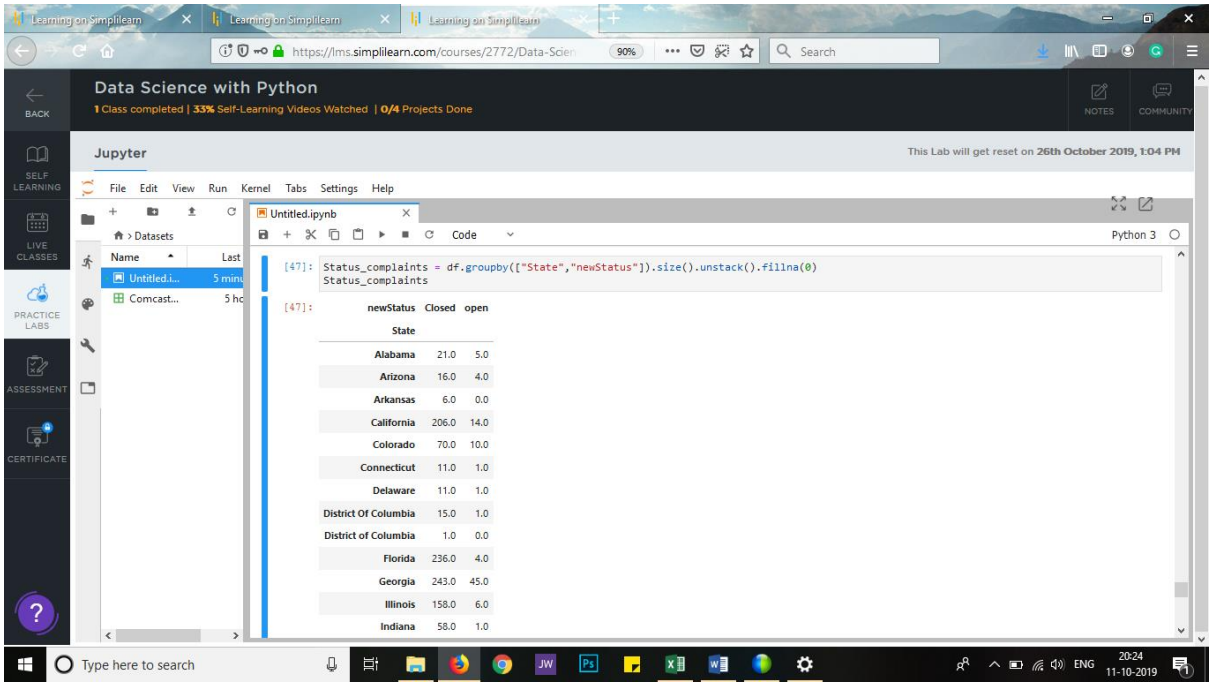
- Which state has the maximum complaints
- Which state has the highest percentage of unresolved complaints

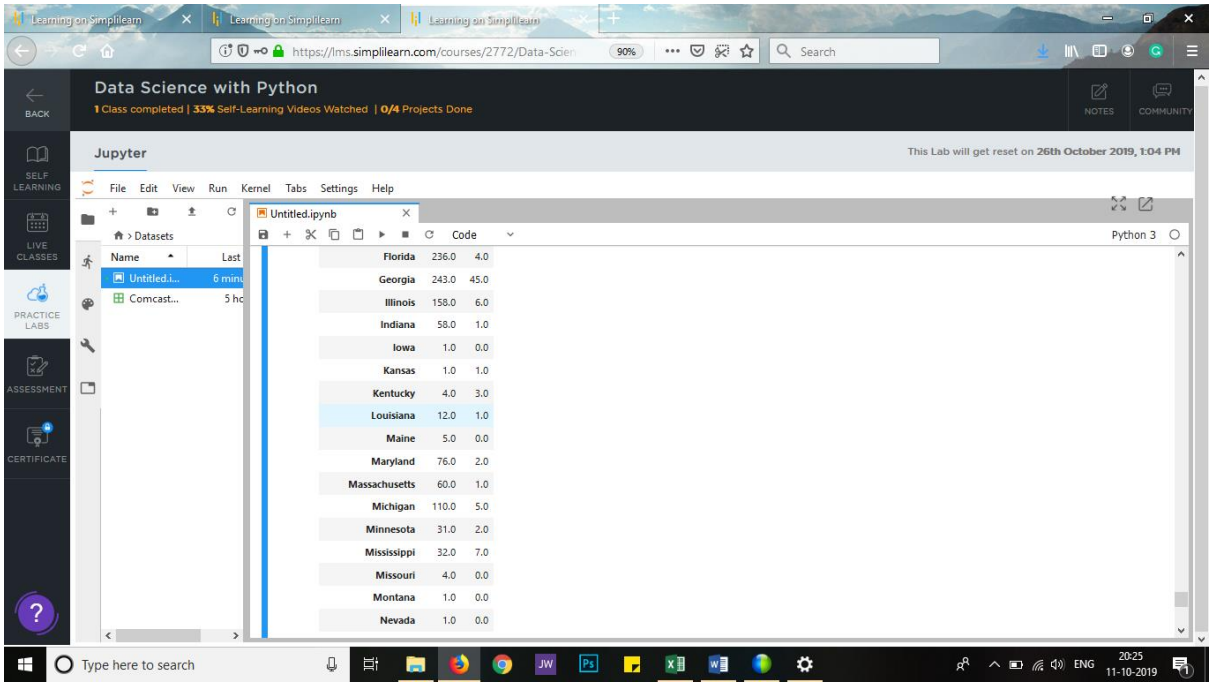
## Code

df.groupby(["State"]).size().sort_values(ascending = False).to_frame().reset_index().rename({0: "Count"} , axis=1)

Status_complaints = df.groupby(["State","newStatus"]).size().unstack().fillna(0)

Status_complaints



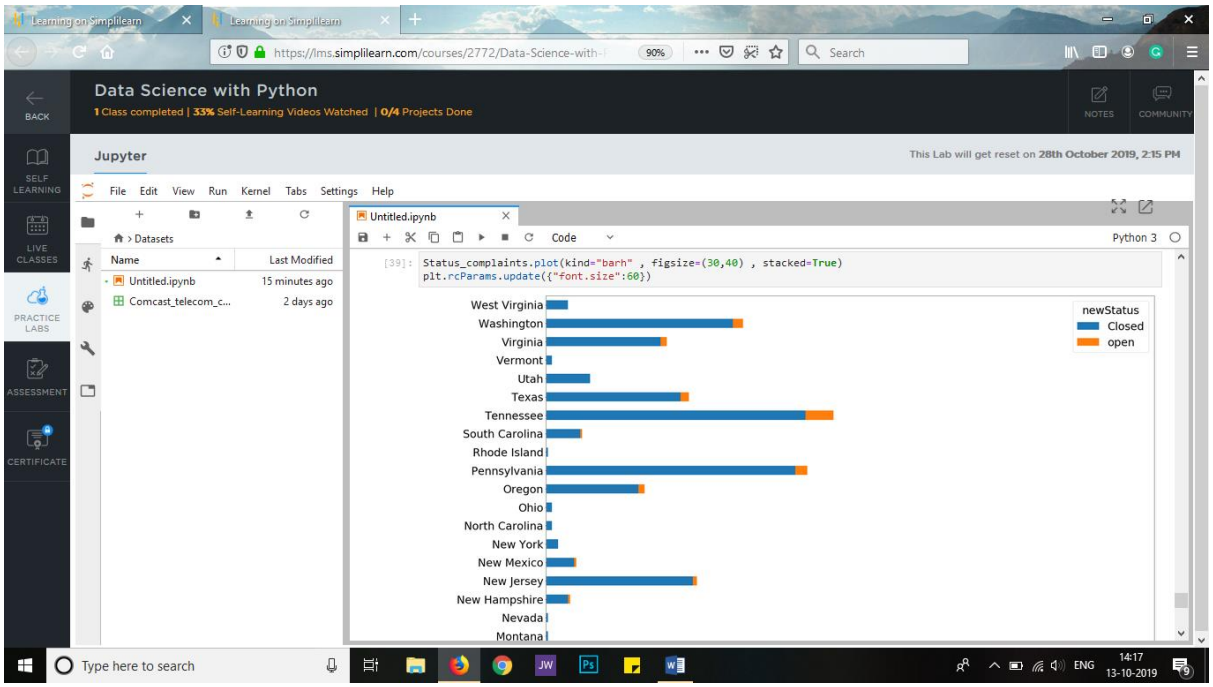

State **Georgia** has maximum complaints

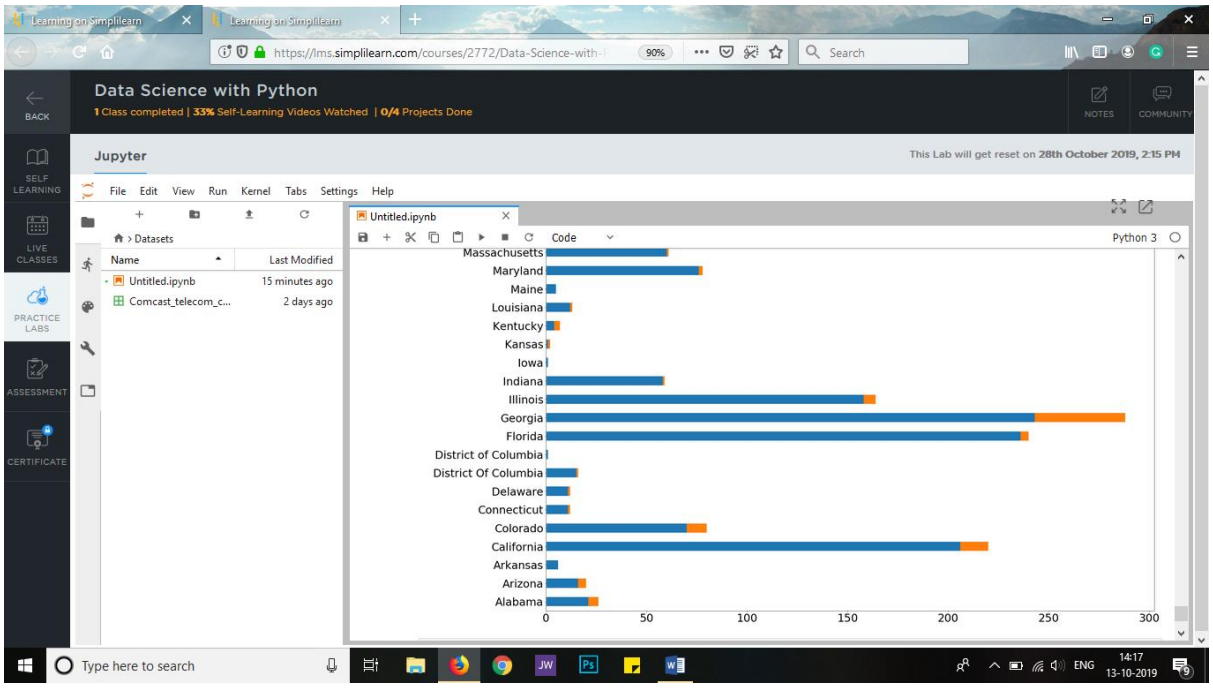
- Complaints : 288
- Unresolved Complaints : 45

# State wise status of complaints in a stacked bar chart

Status_complaints.plot(kind="barh" , figsize=(30,40) , stacked=True)

plt.rcParams.update({"font.size":60})



## Provide the percentage of complaints resolved till date, which were received through the Internet and customer care calls

## Code

import genism

from genism import corpora

dictionary = corpora.Dictionary (doc_clean)

print (dictionary)

doc_term_matrix = [dictionary.doc2bow(doc) for doc in doc_clean]

doc_term_matrix

```
import gensim
from gensim import corpora
```

```
dictionary = corpora.Dictionary(doc_clean)
print(dictionary)
```

```
Dictionary(1412 unique tokens:['cable', 'comcast', 'internet', 'speed', 'disappear']...)
```

```
doc_term_matrix = [dictionary.doc2bow(doc) for doc in doc_clean]
doc_term_matrix
```

```
[[(0, 1), (1, 1), (2, 1), (3, 1)],
 [(4, 1), (5, 1), (6, 1), (7, 1), (8, 1)],
 [(3, 1), (8, 1)],
 [(1, 1), (9, 1), (10, 1), (11, 1), (12, 1), (13, 1), (14, 1), (15, 1)],
 [(1, 1), (8, 1), (16, 1), (17, 1)],
 [(18, 1), (19, 1), (20, 1), (21, 1), (22, 1), (23, 1), (24, 1)],
 [(8, 1), (10, 1), (20, 1), (25, 1), (26, 1)],
 [(1, 1), (8, 1), (27, 1), (28, 1), (29, 1), (30, 1)],
 [(1, 1), (31, 1), (32, 1)],
 [(1, 1), (33, 1), (34, 1), (35, 1), (36, 1)],
 [(5, 1), (8, 1), (37, 1), (38, 1)],
 [(39, 1), (40, 1), (41, 1), (42, 1), (43, 1), (44, 1)],
```

## Code

From genism.models import LdaModel

NUM_TOPICS = 9

Ldamodel = LdaModel (doc_term_matrix, num_topics = NUM_TOPICS , id2word=dictionary , passes = 30)

topics = ldamodel.show_topics()

for topic in topics

```
from gensim.models import LdaModel
```

```
NUM_TOPICS = 9
ldamodel = LdaModel(doc_term_matrix, num_topics=NUM_TOPICS, id2word=dictionary, passes=30)
```

```
topics = ldamodel.show_topics()
for topic in topics
```

```
[(0,
  '0.277*"comcast" + 0.091*"data" + 0.091*"cap" + 0.060*"complaint" + 0.015*"show" + 0.011*"appointment" + 0.010*"slowing" + 0.
010*"rate" + 0.009*"charging" + 0.009*"hbo"'),
 (1,
  '0.094*"comcast" + 0.055*"cable" + 0.054*"bill" + 0.051*"internet" + 0.035*"problem" + 0.033*"without" + 0.029*"month" + 0.02
3*"high" + 0.020*"phone" + 0.017*"email"'),
 (2,
  '0.077*"data" + 0.059*"internet" + 0.055*"comcast" + 0.045*"cap" + 0.031*"overage" + 0.028*"limit" + 0.028*"home" + 0.023*"is
sue" + 0.021*"connectivity" + 0.020*"increased"'),
 (3,
  '0.208*"internet" + 0.143*"service" + 0.116*"comcast" + 0.086*"speed" + 0.027*"slow" + 0.023*"poor" + 0.021*"throttling" + 0.
011*"paying" + 0.011*"bill" + 0.009*"back"'),
 (4,
  '0.110*"comcast" + 0.066*"internet" + 0.065*"xfinity" + 0.061*"data" + 0.045*"cap" + 0.033*"usage" + 0.027*"false" + 0.027*"d
eceptive" + 0.022*"business" + 0.020*"switch"'),
 (5,
  '0.124*"comcast" + 0.088*"charge" + 0.054*"price" + 0.044*"fee" + 0.027*"contract" + 0.022*"monopoly" + 0.022*"fraudulent" +
```

# Code

word_dict = {}

for i in range(NUM_TOPICS):

words = ldamodel.show_topics(i,topic = 20)

word_dict[“Topic # “ + “{}”.format(i+1)] = [i[0] for i in words ]

pd.DataFrame(word_dict)

```
word_dict = {}
for i in range(NUM_TOPICS):
    words = ldamodel.show_topic(i, topn = 20)
    word_dict["Topic # " + "{}".format(i+1)] = [i[0] for i in words]
```

```
pd.DataFrame(word_dict)
```

| | Topic # 1 | Topic # 2 | Topic # 3 | Topic # 4 | Topic # 5 | Topic # 6 | Topic # 7 | Topic # 8 | Topic # 9 |
|---|---|---|---|---|---|---|---|---|---|
| 0 | comcast | comcast | data | internet | comcast | comcast | service | speed | billing |
| 1 | data | cable | internet | service | internet | charge | comcast | pay | comcast |
| 2 | cap | bill | comcast | comcast | xfinity | price | day | promised | service |
| 3 | complaint | internet | cap | speed | data | fee | account | access | issue |
| 4 | show | problem | overage | slow | cap | contract | bill | time | practice |
| 5 | appointment | without | limit | poor | usage | monopoly | billed | several | customer |
| 6 | slowing | month | home | throttling | false | fraudulent | refund | mb | unfair |
| 7 | rate | high | issue | paying | deceptive | year | call | plan | pricing |
| 8 | charging | phone | connectivity | bill | business | cramming | lack | low | complaint |
| 9 | hbo | email | increased | back | switch | modem | shitty | scam | comcastxfinity |
| 10 | go | ps4 | monthly | customer | advertising | payment | 12 | disconnection | monopolistic |
| 11 | 300gb | provider | charged | outage | intermittent | connection | loss | wont | terrible |

ldamodel.show_topic(0, topn = 20)

```
ldamodel.show_topic(0, topn = 20)
```

```
[('comcast', 0.27682346),
 ('data', 0.09102981),
 ('cap', 0.090602614),
 ('complaint', 0.059584655),
 ('show', 0.014601116),
 ('appointment', 0.011381063),
 ('slowing', 0.010168295),
 ('rate', 0.0097776),
 ('charging', 0.008840713),
 ('hbo', 0.008578849),
 ('go', 0.008174605),
 ('300gb', 0.008160758),
 ('overcharge', 0.008160724),
 ('3', 0.007352107),
 ('information', 0.007345246),
 ('bill', 0.00673711),
 ('blocking', 0.0065576984),
 ('credit', 0.0065480573),
 ('outage', 0.006494935),
 ('consumer', 0.0061397543)]
```

# Code

Import pyLDAvis.gensim

Lda_display = pyLDAvis.gesim.prepare (ldamodel, doc_term_matrix, dictionary , sort_topics = False )

PyLDAvis.display (Lda_display)