Semester	T.E. Semester V – Computer Engineering
Subject	Data Warehousing and Mining
Subject Professor In-charge	Prof. Kavita Shirsat
Assisting Teachers	Prof. Kavita Shirsat
Laboratory	M-313A

Student Name	Vibodh Bhosure	
Roll Number	20102A0032	
Grade and Subject Teacher's Signature		

Experiment Number	02	
Experiment Title	To implement data visualization for a given dataset.	
Resources / Apparatus Required	Hardware: Computer system	Software: Python
Description	• Data visualization is a general term that describes any effort to help people understand the significance of data by placing it in a visual context. Patterns, trends, and correlations that might go undetected in text-based data can be exposed and recognized easier with data visualization software. • Today's data visualization tools go beyond the standard charts and graphs used in Microsoft Excel spreadsheets, displaying data in more sophisticated ways such as infographics, dials and gauges, geographic maps, sparklines, heat maps, and detailed bar, pie, and fever charts. • The images may include interactive capabilities, enabling users to manipulate them or drill into the data for querying and analysis. Indicators designed to alert users when data has been updated or predefined conditions occur can also be included.	
Program		

```
#!/usr/bin/env python
# coding: utf-8

# In[1]:


import pandas as pd
import numpy as np


# In[2]:
```

```
import matplotlib
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
import plotly.graph_objects as go
from plotly.subplots import make_subplots
from matplotlib import cm
import missingno as msno


# In[3]:


get_ipython().run_line_magic('matplotlib', 'inline')
sns.set_style('darkgrid')
matplotlib.rcParams['figure.facecolor'] =
'#00000000'
matplotlib.rcParams['figure.facecolor'] =
'#00000000'


# In[5]:


import os
from wordcloud import WordCloud

import warnings
warnings.filterwarnings("ignore")


# In[7]:


df = pd.read_csv('udemy_courses.csv')
df.sample(5).reset_index(drop=True)


# In[8]:


df.columns


# In[9]:


df.drop(['course_title','url'], axis=1,
inplace=True)


# In[11]:
```

```
df.sample(5).reset_index(drop=True)


# In[14]:


df['published_timestamp'] =
pd.to_datetime(df['published_timestamp'])


# In[16]:


df['year'] = df['published_timestamp'].dt.year


# In[17]:


df['content_duration'] =
(df['content_duration']*60).astype(int)


# In[19]:


df.duplicated().sum()


# In[21]:


df.isnull().sum()


# In[22]:


msno.matrix(df)
plt.title('Distribution of Missing Values');


# In[23]:


from IPython.core.display import HTML

#Acceps a list of IpyTable objects and returns a
table which contains each IpyTable in a cell
def multi_table(table_list):
        return HTML('<table><tr style="background-
color:white;">' +  ''.join(['<td>' +
```

```
table._repr_html_() + '</td>' for table in
table_list]) +'</tr></table>')


# In[24]:


nunique_df={var:pd.DataFrame(df[var].value_counts())
           for var in {'is_paid',
'level','subject'}}

multi_table([nunique_df['is_paid'],nunique_df['level
'],nunique_df['subject']])


# ## Popularity of course subjects

# In[25]:


df['tmp'] = 1
fig = px.pie(df, names='subject',values='tmp',hole =
0.6,title='relation tips')
fig.update_traces(textposition='outside',
textinfo='percent+label')
fig.update_layout(
    title_text="Subject percentage",
    annotations=[dict(text='course subjects', x=0.5,
y=0.5, font_size=20, showarrow=False)])


# ### Subjects Business Finance and Web Development
have high number of courses

# ## Popularity of level

# In[27]:


df['tmp'] = 1
fig = px.pie(df, names='level',values='tmp',hole =
0.6,title='relation tips')
fig.update_traces(textposition='outside',
textinfo='percent+label')
fig.update_layout(
    title_text="Level Percentage",
    annotations=[dict(text='course levels', x=0.5,
y=0.5, font_size=20, showarrow=False)])


# ### Most of the courses are for all level

# In[28]:
```

```
subject_by_year=pd.pivot_table(df, index='year',
columns=['subject'], values='course_id',
aggfunc='count')
subject_by_year.fillna(0, inplace=True)
subject_by_year.style.set_properties(**{'background-
color': '#F2C4EE ' ,'color':'black','border-color':
'#8b8c8c'})
fig, axs = plt.subplots(2,2, figsize=(13,5))
ind = 0
for i in range(2):
    for j in range(2):
        sns.lineplot(x=subject_by_year.index,
y=subject_by_year.iloc[:,ind], ax=axs[i,j])
        axs[i,j].text(2016.7,subject_by_year.iloc[-
1,ind]-20,int(subject_by_year.iloc[-1,ind]))
        ind +=1
plt.suptitle('Subjects / Change Over Years')
plt.tight_layout();


# In[32]:


subject_year = df.groupby(['year','subject']).size()
subject_2011 =
np.round(subject_year[2011].values/subject_year[2011
].values.sum(),2)
subject_2012 =
np.round(subject_year[2012].values/subject_year[2012
].values.sum(),2)
subject_2013 =
np.round(subject_year[2013].values/subject_year[2013
].values.sum(),2)
subject_2014 =
np.round(subject_year[2014].values/subject_year[2014
].values.sum(),2)
subject_2015 =
np.round(subject_year[2015].values/subject_year[2015
].values.sum(),2)
subject_2016 =
np.round(subject_year[2016].values/subject_year[2016
].values.sum(),2)
subject_2017 =
np.round(subject_year[2017].values/subject_year[2017
].values.sum(),2)
fig = go.Figure()
categories = ['Business Finance', 'Graphic
Design','Musical Instruments','Web Development']
fig.add_trace(go.Scatterpolar(
            r = subject_2011,
            theta = categories,
            fill = 'toself',
            name = '2011 course subject'
```

```
            ))
fig.add_trace(go.Scatterpolar(
            r = subject_2012,
            theta = categories,
            fill = 'toself',
            name = '2012 course subject'
#             fillcolor = 'lightred'
            ))
fig.add_trace(go.Scatterpolar(
            r = subject_2013,
            theta = categories,
            fill = 'toself',
            name = '2013 course subject'
#             fillcolor = 'lightblue'
            ))
fig.add_trace(go.Scatterpolar(
            r = subject_2014,
            theta = categories,
            fill = 'toself',
            name = '2014 course subject'
#             fillcolor = 'lightblue'
            ))
fig.add_trace(go.Scatterpolar(
            r = subject_2015,
            theta = categories,
            fill = 'toself',
            name = '2015 course subject'
#             fillcolor = 'lightblue'
            ))
fig.add_trace(go.Scatterpolar(
            r = subject_2016,
            theta = categories,
            fill = 'toself',
            name = '2016 course subject'
#             fillcolor = 'lightblue'
            ))
fig.add_trace(go.Scatterpolar(
            r = subject_2017,
            theta = categories,
            fill = 'toself',
            name = '2017 course subject'
#             fillcolor = 'lightblue'
            ))
fig.update_layout(
  polar=dict(
    radialaxis=dict(
#       visible=True,
      range=[0, 0.75]
    )),
  font = dict(family="Franklin Gothic", size=17),
  showlegend=True,
  title = 'Rate of course subject by year'
)
fig.layout.template = 'plotly_dark'
```

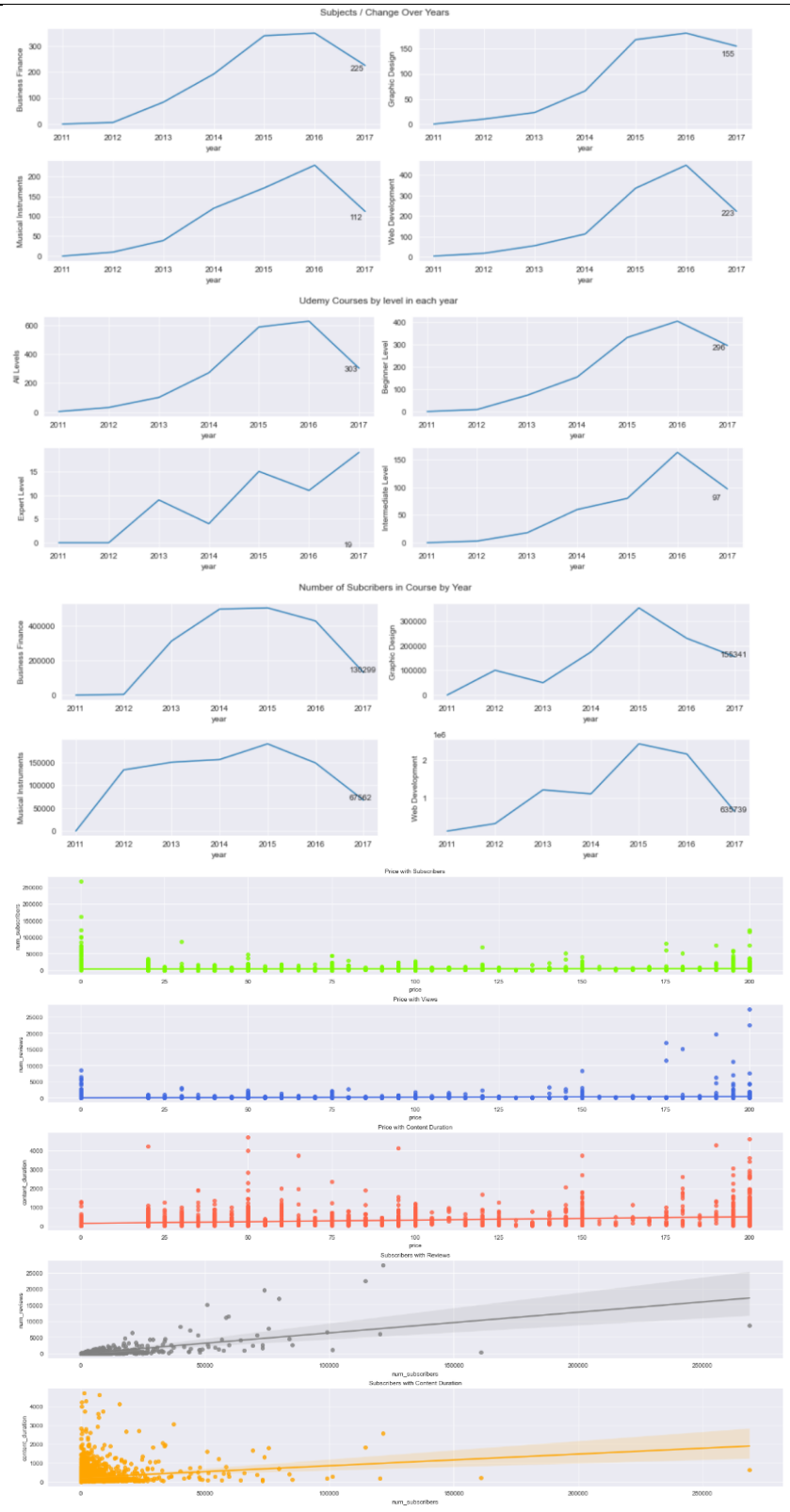
```
fig.show()


# In[33]:


level_by_year=pd.pivot_table(df, index='year',
columns=['level'], values='course_id',
aggfunc='count')
level_by_year.fillna(0, inplace=True)
fig, axs = plt.subplots(2,2, figsize=(13,5))
ind = 0
for i in range(2):
    for j in range(2):
        sns.lineplot(x=level_by_year.index,
y=level_by_year.iloc[:,ind], ax=axs[i,j])
        axs[i,j].text(2016.7,level_by_year.iloc[-
1,ind]-20,int(level_by_year.iloc[-1,ind]))
        ind +=1
plt.suptitle('Udemy Courses by level in each year')
plt.tight_layout();


# In[35]:


level_by_year=df.groupby('year')['level'].value_coun
ts().reset_index(level=0).rename(columns={'level':'l
evel count'}, index={'index':'Level_of_Courses'})
level_by_year
fig=px.line(level_by_year, x='year', y='level
count', color=level_by_year.index, title='Udemy
Courses by level in each year')
fig.show()


# In[36]:


nsub_by_year=pd.pivot_table(df, index='year',
columns=['subject'], values='num_subscribers',
aggfunc='sum')
nsub_by_year.fillna(0, inplace=True)
fig, axs = plt.subplots(2,2, figsize=(13,5))
ind = 0
for i in range(2):
    for j in range(2):
        sns.lineplot(x=nsub_by_year.index,
y=nsub_by_year.iloc[:,ind], ax=axs[i,j])
        axs[i,j].text(2016.7,nsub_by_year.iloc[-
1,ind]-20,int(nsub_by_year.iloc[-1,ind]))
        ind +=1
plt.suptitle('Number of Subcribers in Course by
Year')
```

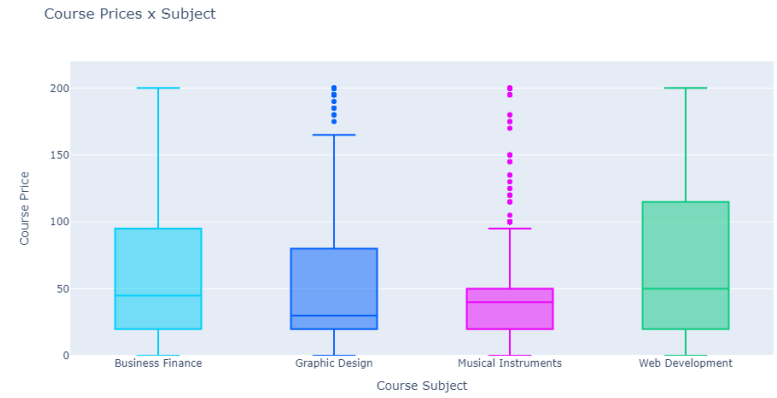
```
plt.tight_layout();


# In[37]:


fig=px.box(df,
           x='content_duration',
           y='is_paid',
           orientation='h',
           color='is_paid',
           title='Duration Distribution Across Type
of Course (charge or free)',

color_discrete_sequence=['#8ACE12','#AF85D2']
           )

fig.update_layout(showlegend=False)
fig.update_xaxes(title='Content Duration')
fig.update_yaxes(title='Paid Course')
fig.show()


# In[38]:


fig=px.box(df,
           x='content_duration',
           y='subject',
           orientation='h',
           color='is_paid',
           title='Duration Distribution Across Type
of Course (subject)',

color_discrete_sequence=['#8ACE12','#AF85D2']
           )

fig.update_layout(showlegend=False)
fig.update_xaxes(title='Content Duration')
fig.update_yaxes(title='Course Subject')
fig.show()


# In[39]:


def pltplot(data, xcol, ycol,color, ax, title):
    sns.regplot(data=data, x=xcol, y=ycol,
color=color, ax=ax).set_title(title, size=10)


# In[40]:
```

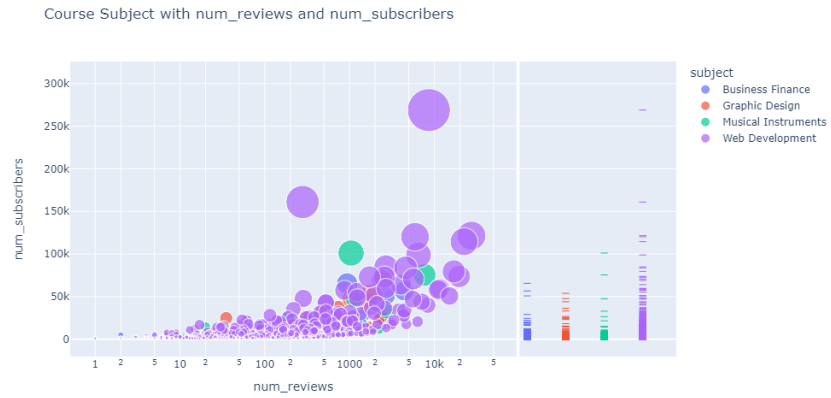
```
fig, ((ax1),(ax2),(ax3), (ax4),
(ax5))=plt.subplots(ncols=1, nrows=5)
fig.set_size_inches(18,15)
fig.tight_layout(pad=3.0)

pltplot(df,
'price','num_subscribers','lawngreen',ax1, 'Price
with Subscribers')
pltplot(df, 'price','num_reviews','royalblue', ax2,
'Price with Views')
pltplot(df, 'price','content_duration','tomato',
ax3, 'Price with Content Duration')
pltplot(df, 'num_subscribers','num_reviews','gray',
ax4,'Subscribers with Reviews')
pltplot(df,
'num_subscribers','content_duration','orange',
ax5,'Subscribers with Content Duration')


# In[41]:


fig = px.scatter(df, x="num_reviews",
y="num_subscribers",
                 size="num_subscribers",
color="subject",
                 log_x=True, size_max=50,
                 title="Course Subject with
num_reviews and num_subscribers",
                 marginal_y='rug')
fig.show()


# In[42]:


paid_courses_df = df.query("price != 'Free'")
paid_courses_df['price'] =
df['price'].astype('float32')
fig = px.box(paid_courses_df,
             x     = 'subject',
             y     = 'price',
             color = 'subject',
             title = 'Course Prices x Subject',
             color_discrete_sequence =
['#03cffc','#0362fc','#eb03fc','#0ecc83'],
            )

fig.update_layout(showlegend=False)
fig.update_yaxes(range=[0,220], title='Course
Price')
fig.update_xaxes(title='Course Subject')
fig.show()
```

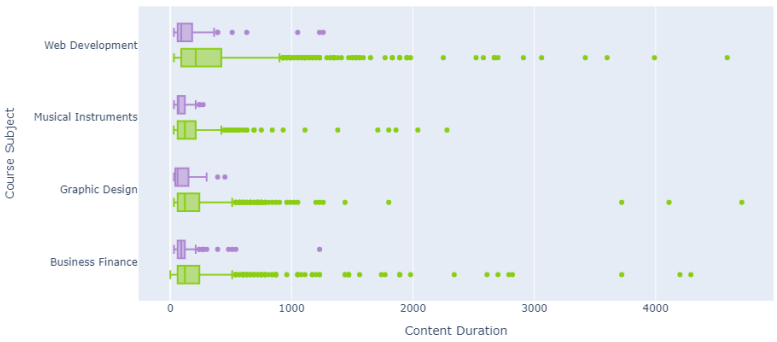
```
# In[ ]:
```

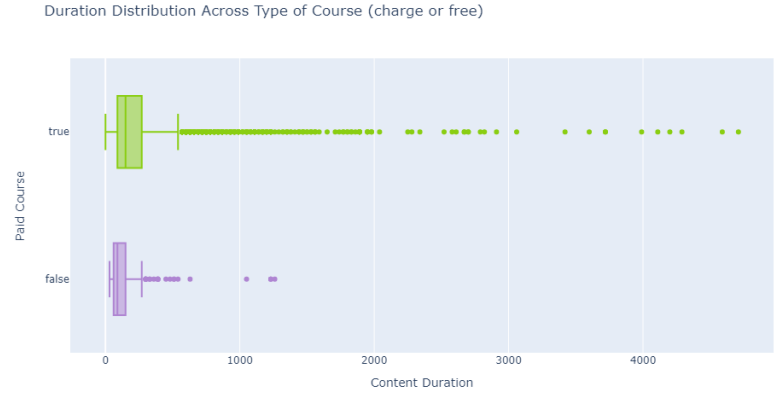
Output

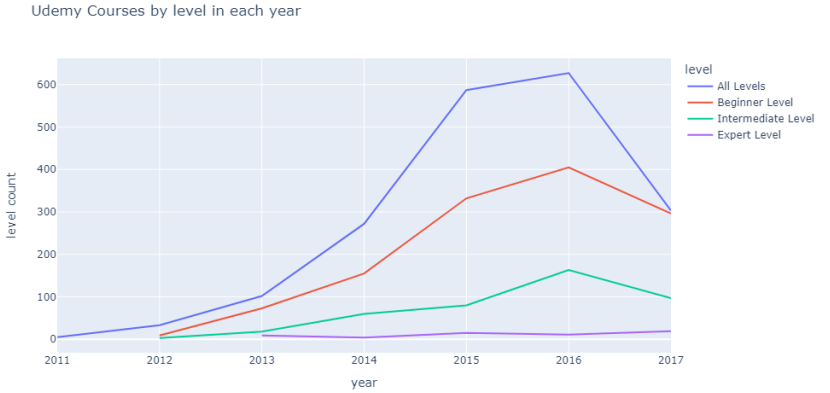

Subjects / Change Over Years
Business Finance
Graphic Design
Musical Instruments
Web Development
year
Udemy Courses by level in each year
All Levels
Beginner Level
Expert Level
Intermediate Level
Number of Subcribers in Course by Year
Price with Subscribers
Price with Views
Price with Content Duration
Subscribers with Reviews
Subscribers with Content Duration
num_subscribers
num_reviews
content_duration
price

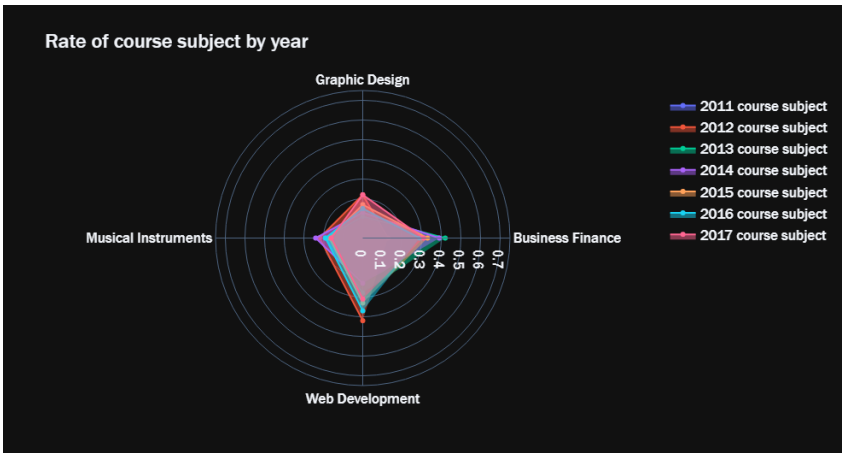

Course Prices x Subject
Course Price
200
150
100
50
0
Business Finance
Graphic Design
Musical Instruments
Web Development
Course Subject

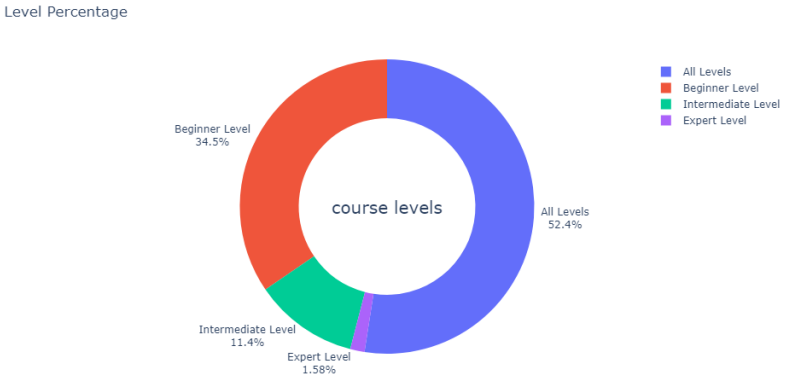

Course Subject with num_reviews and num_subscribers
num_subscribers
300k
250k
200k
150k
100k
50k
0
1
2
5
10
2
5
100
2
5
1000
2
5
10k
2
5
num_reviews
subject
Business Finance
Graphic Design
Musical Instruments
Web Development

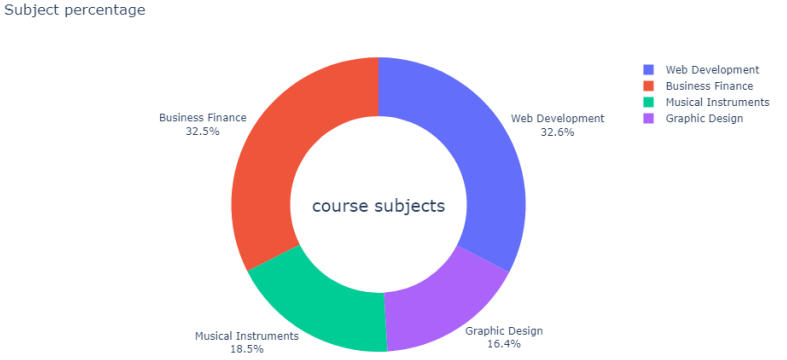

Duration Distribution Across Type of Course (subject)
Course Subject
Web Development
Musical Instruments
Graphic Design
Business Finance
0
1000
2000
3000
4000
Content Duration


Duration Distribution Across Type of Course (charge or free)
Paid Course
true
false
0
1000
2000
3000
4000
Content Duration









Conclusion:	• Data visualization is a general term that describes any effort to help people understand the significance of data by placing it in a visual context. Patterns, trends, and correlations

	that might go undetected in text-based data can be exposed and recognized easier with data visualization software. • Today's data visualization tools go beyond the standard charts and graphs used in Microsoft Excel spreadsheets, displaying data in more sophisticated ways such as infographics, dials and gauges, geographic maps, sparklines, heat maps, and detailed bar, pie, and fever charts. • The images may include interactive capabilities, enabling users to manipulate them or drill into the data for querying and analysis. Indicators designed to alert users when data has been updated or predefined conditions occur can also be included.