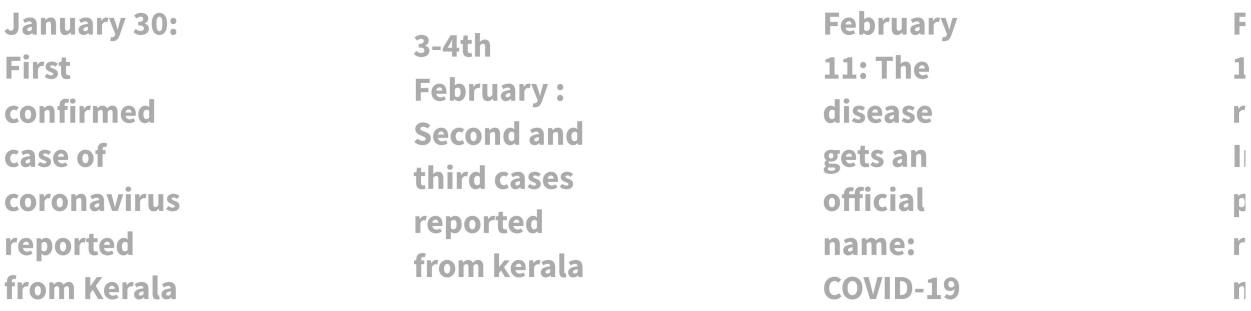
Viewers accessing the notebook through github are requested to view the notebook on google colaboratory because the outputs of the following notebook are not supported by github notebook viewer.

Timeline

```
from IPython.core.display import HTML
HTML('''<div class="flourish-embed flourish-cards" data-src="visualisation/1786965" data-url=
```



Interactive content by Flourish

Loading Libraries

```
!pip install pycountry
!pip install plotly==4.5.2
```

```
# visualization
import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd
import numpy as np
import pycountry
import folium
from datetime import datetime
from datetime import timedelta
```

```
import plotly.express as px
import plotly.graph_objects as go
from plotly.subplots import make_subplots
# hide warnings
import warnings
warnings.filterwarnings('ignore')


# color pallette
cnf = '#393e46' # confirmed - grey
dth = '#ff2e63' # death - red
rec = '#21bf73' # recovered - cyan
act = '#fe9801' # active case - yellow
```

Loading Datasets

```
AgeGroupDetails = pd.read_csv("/content/drive/My Drive/Colab Notebooks/Corona India/AgeGroupD
AgeGroupDetails.head(2)
```

	Sno	AgeGroup	TotalCases	Percentage
0	1	0-9	22	3.18%
1	2	10-19	27	3.90%

```
p_df = pd.read_csv("/content/drive/My Drive/Colab Notebooks/Corona India/patients_data.csv")
# fixing date format
p_df['date_announced'] = pd.to_datetime(p_df['date_announced'], format='%d/%m/%Y')
p_df['status_change_date'] = pd.to_datetime(p_df['status_change_date'], format='%d/%m/%Y')

# fix nationality values
p_df['nationality'] = p_df['nationality'].replace('Indian', 'India')

p_df.head(3)
```

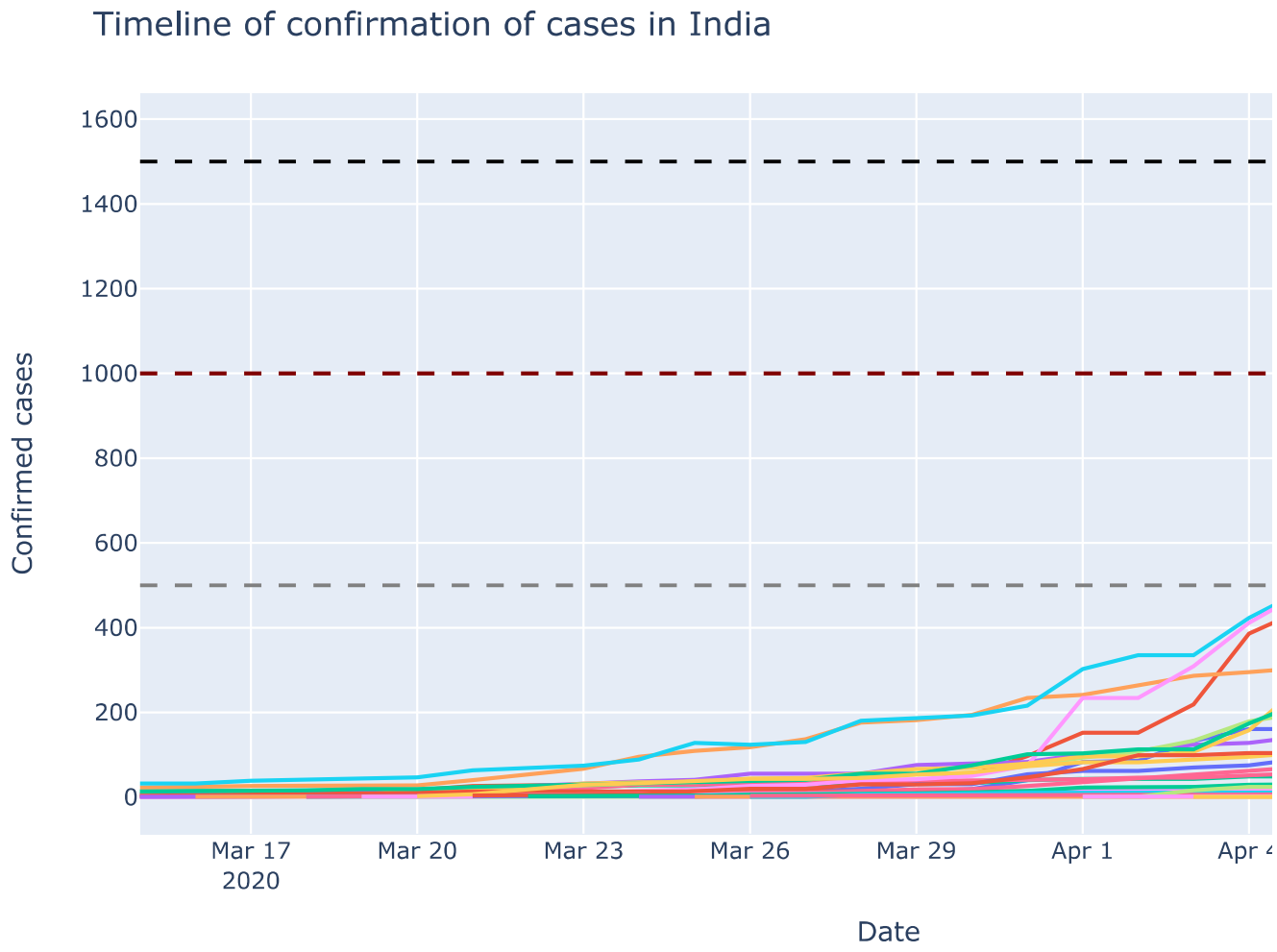
```
        patient_number  state_patient_number  date_announced  age_bracket  gender  detected_
df = pd.read_csv("/content/drive/My Drive/Colab Notebooks/Corona India/complete.csv",parse_da
# replace 'union territory' with ''
df['Name of State / UT'] = df['Name of State / UT'].str.replace('Union Territory of ', '')

# rearrange columns
df = df[['Date', 'Name of State / UT', 'Latitude', 'Longitude', 'Total Confirmed cases', 'Dea

# rename columns
df.columns = ['Date', 'State/UT', 'Latitude', 'Longitude', 'Confirmed', 'Deaths', 'Cured']

# fix datatype
for i in ['Confirmed', 'Deaths', 'Cured']:
    df[i] = df[i].astype('int')

# derived columns
df['Active'] = df['Confirmed'] - df['Deaths'] - df['Cured']
df['Mortality rate'] = df['Deaths']/df['Confirmed']
df['Recovery rate'] = df['Cured']/df['Confirmed']

# rearrange columns
df = df[['Date', 'State/UT', 'Latitude', 'Longitude', 'Confirmed', 'Active', 'Deaths', 'Morta

# final data
df.head()
```

	Date	State/UT	Latitude	Longitude	Confirmed	Active	Deaths	Mortality rate	Cu
0	2020-01-30	Kerala	10.8505	76.2711	1	1	0	0.0	
1	2020-01-31	Kerala	10.8505	76.2711	1	1	0	0.0	
2	2020-02-01	Kerala	10.8505	76.2711	2	2	0	0.0	

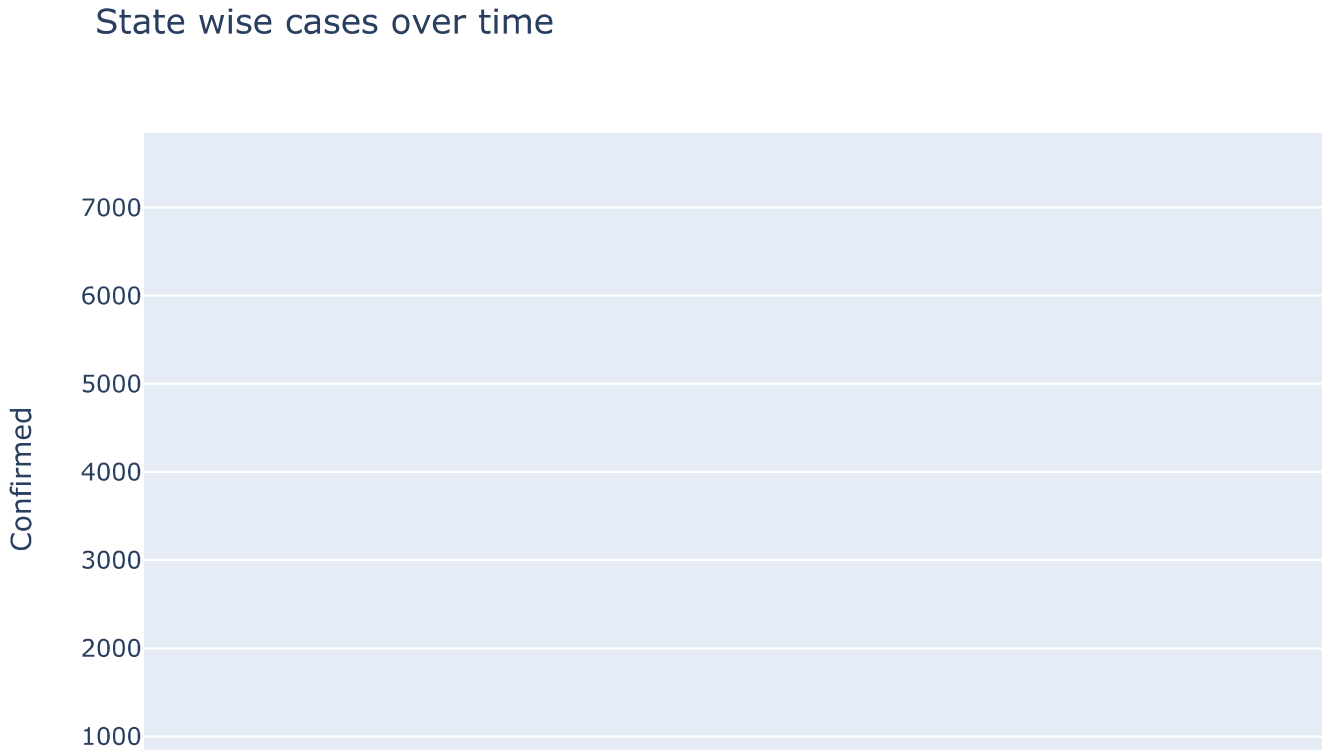
Confirmation of cases in states over time

```
temp = df[df['Date']>='2020-03-15']
fig = px.line(temp,x='Date', y='Confirmed',color='State/UT')
fig.update_layout(title='Timeline of confirmation of cases in India',
                  xaxis_title='Date', yaxis_title='Confirmed cases')

fig.add_trace(go.Scatter(x=temp['Date'], y=[500]*len(temp),
                         mode='lines', name='500 count line',
                         line = dict(dash='dash',color='gray')))
```

```
fig.add_trace(go.Scatter(x=temp['Date'], y=[1000]*len(temp),
                         mode='lines', name='1000 count line',
                         line = dict(dash='dash',color='maroon')))

fig.add_trace(go.Scatter(x=temp['Date'], y=[1500]*len(temp),
                         mode='lines', name='1000 count line',
                         line = dict(dash='dash',color='black')))
fig.show()
```

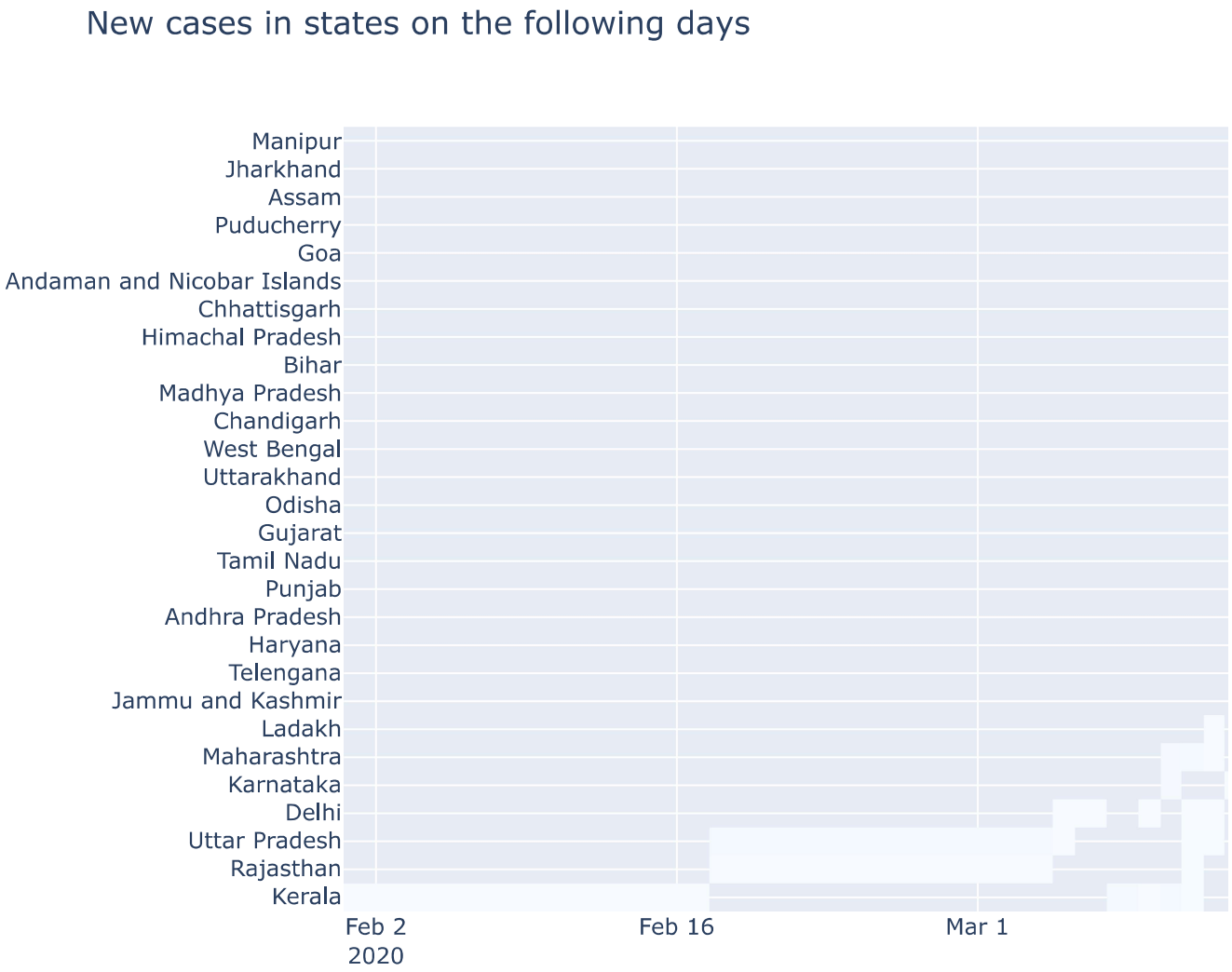



```
fig = px.bar(df.sort_values('Confirmed', ascending=False), x="Date", y="Confirmed", color='St
             color_discrete_sequence = px.colors.qualitative.Vivid)
fig.update_traces(textposition='outside')
fig.update_layout(uniformtext_minsize=8, uniformtext_mode='hide')
fig.show()
```

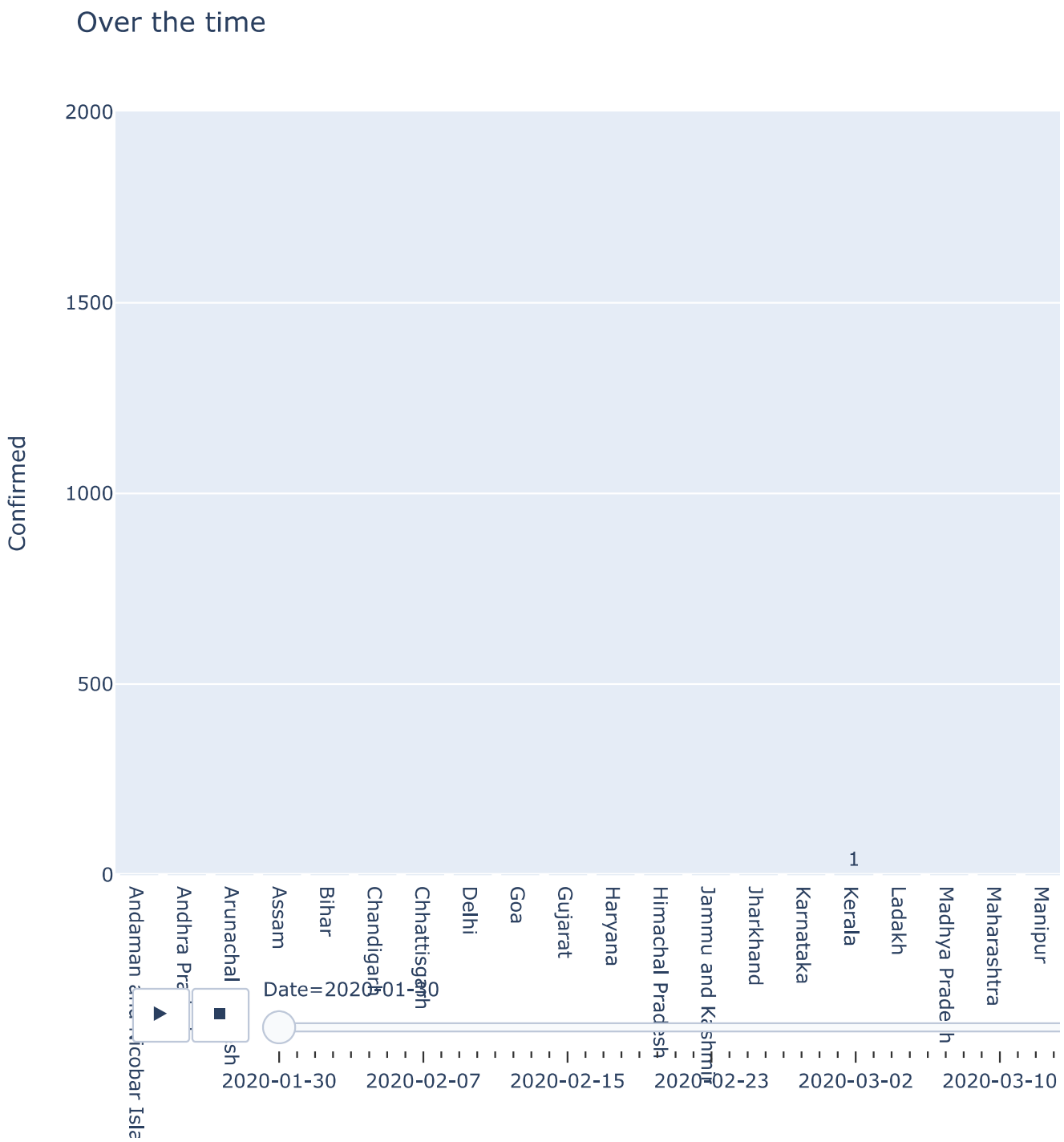



```
mask = (temp['State/UT'] != temp['State/UT'].shift(1))
temp.loc[mask, 'Confirmed'] = np.nan

temp = df.groupby(['State/UT', 'Date',])['Confirmed'].sum().diff().reset_index()
# diff helps in calculating diff between second and first row for all rows

# renaming columns
temp.columns = ['State/UT', 'Date', 'New cases']
temp = temp.dropna()
temp = temp[temp['New cases']>0]
temp = temp.sort_values(['Date', 'State/UT']).reset_index(drop=True)

# Snippet to understand shift and mask

# a = pd.DataFrame({'a':[1,1,1,1],'b':[1,2,3,4]})
# mask = a==a.shift(1)
# mask


fig = go.Figure(data=go.Heatmap(
        z=temp['New cases'],
        x=temp['Date'],
        y=temp['State/UT'],
        colorscale='Blues',
        showlegend=False,
        text=temp['New cases']))

fig.update_layout(yaxis = dict(dtick = 1))
fig.update_layout(title='New cases in states on the following days')
```

```
fig.update_layout(height=600)
fig.show()
```



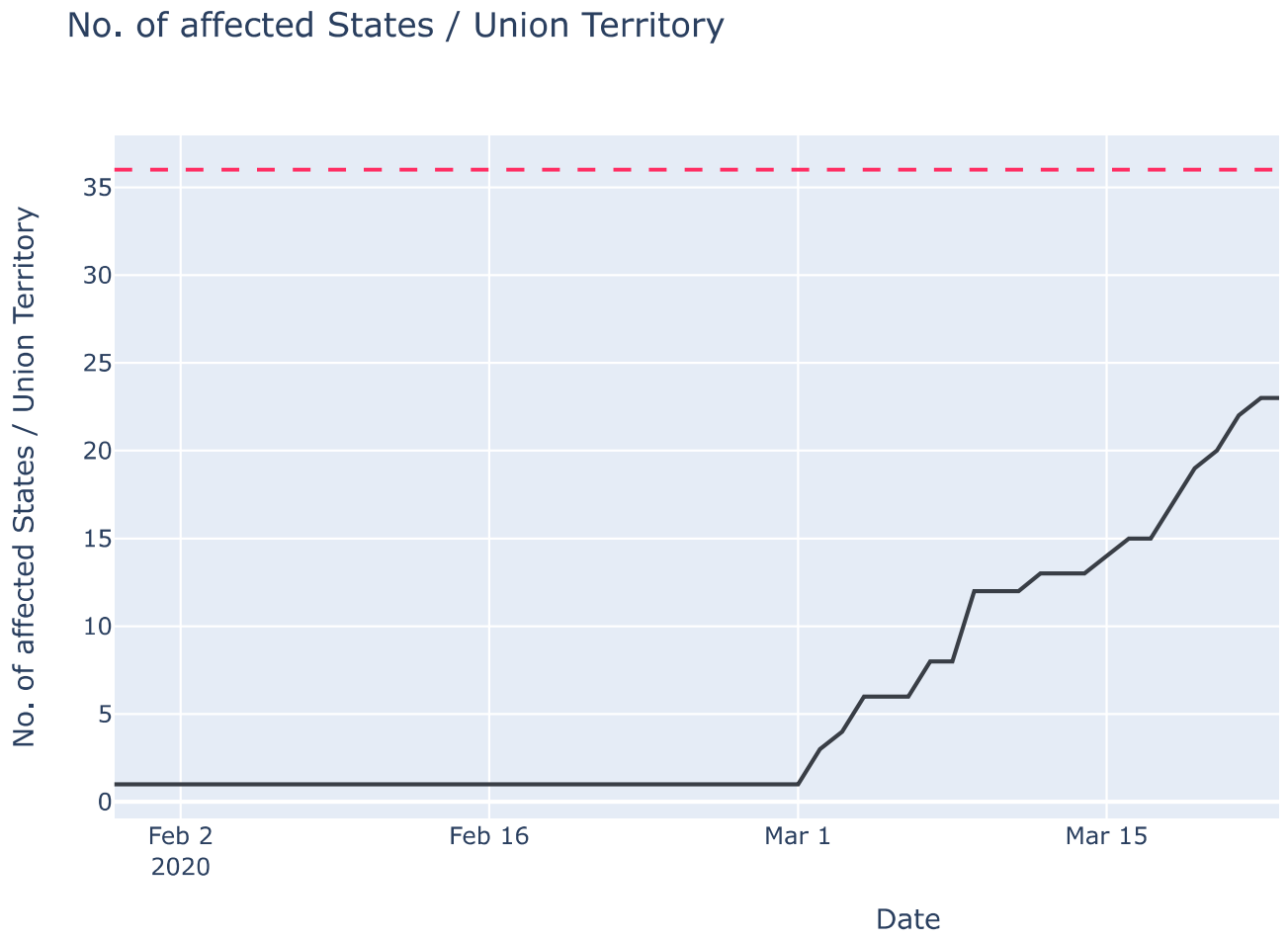

```
temp = df.copy(deep=True)
temp['Date'] = temp['Date'].dt.strftime('%Y-%m-%d')
temp = temp.pivot(index='Date', columns='State/UT', values='Confirmed').fillna(0).reset_index
temp = temp.melt(id_vars='Date', value_name='Confirmed')

fig = px.bar(temp, y='Confirmed', x='State/UT', color='Confirmed',
             text='Confirmed', title='Over the time', animation_frame='Date',
             range_y=[0,2000],hover_data=['Date','Confirmed','State/UT'],color_continuous_sca
fig.show()
```



History of the count of affected states with time

```
no_of_states = df.groupby('Date')['State/UT'].unique().apply(len).values
dates = df.groupby('Date')['State/UT'].unique().apply(len).index

fig = go.Figure()

fig.add_trace(go.Scatter(x=dates, y=[36 for i in range(len(no_of_states))],
                         mode='lines', name='Total States+UT',
```

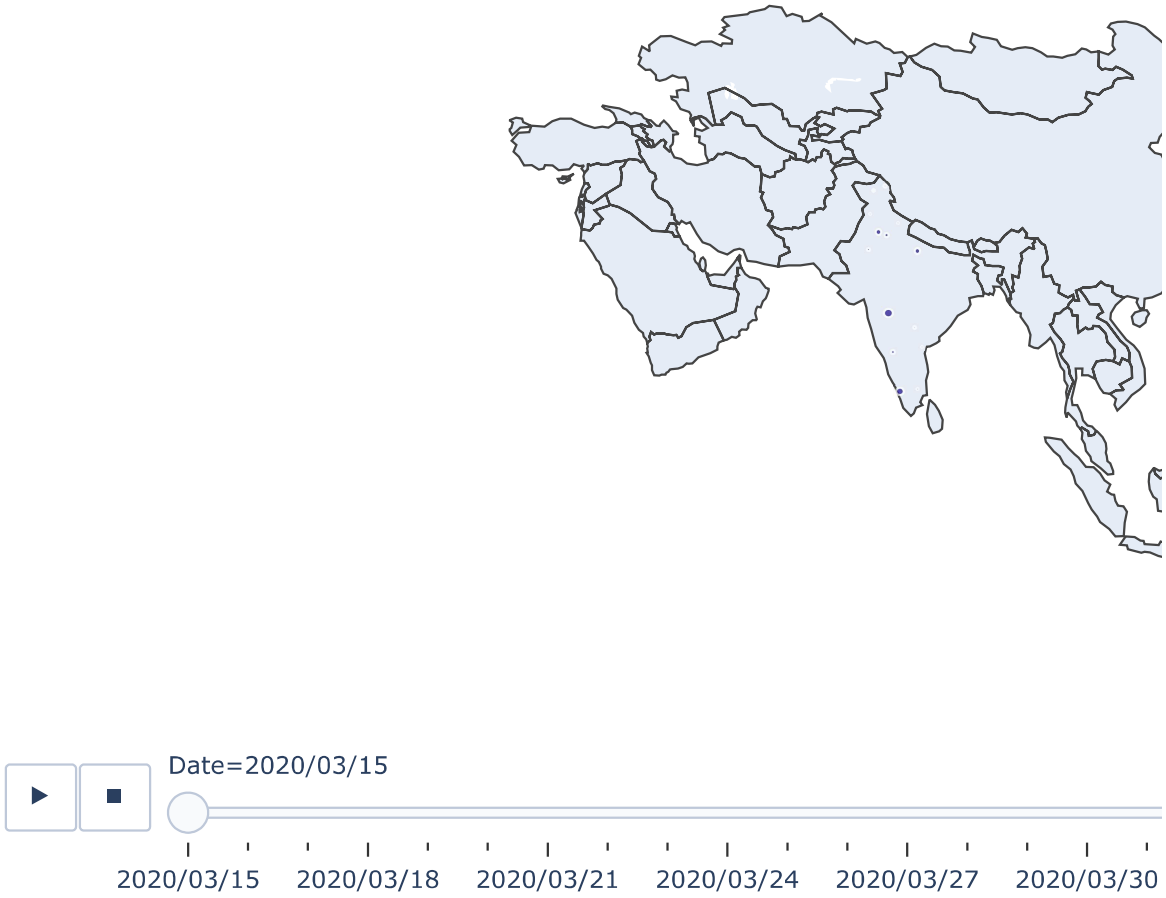
```
                        line = dict(dash='dash',color=dth)))

fig.add_trace(go.Scatter(x=dates, y=no_of_states, hoverinfo='x+y',
                         mode='lines', name='No. of affected States+UT',
                         line = dict(color=cnf)))

fig.update_layout(title='No. of affected States / Union Territory',
                  xaxis_title='Date', yaxis_title='No. of affected States / Union Territory')
fig.update_traces(textposition='top center')
fig.update_layout(uniformtext_minsize=8, uniformtext_mode='hide')
fig.show()
```



Visualisation of the spread through maps

```
temp = df.copy()
temp['Date'] = temp['Date'].dt.strftime('%Y/%m/%d')
temp = temp[temp['Date']>='2020/03/15']
fig = px.scatter_geo(temp, lat="Latitude", lon="Longitude", color='Confirmed', size='Confirme
                     hover_name="State/UT", scope='asia', animation_frame="Date", center={'la
```

```
                      range_color=[0, max(temp['Confirmed'])])
fig.show()
```



Analysis of the latest numbers

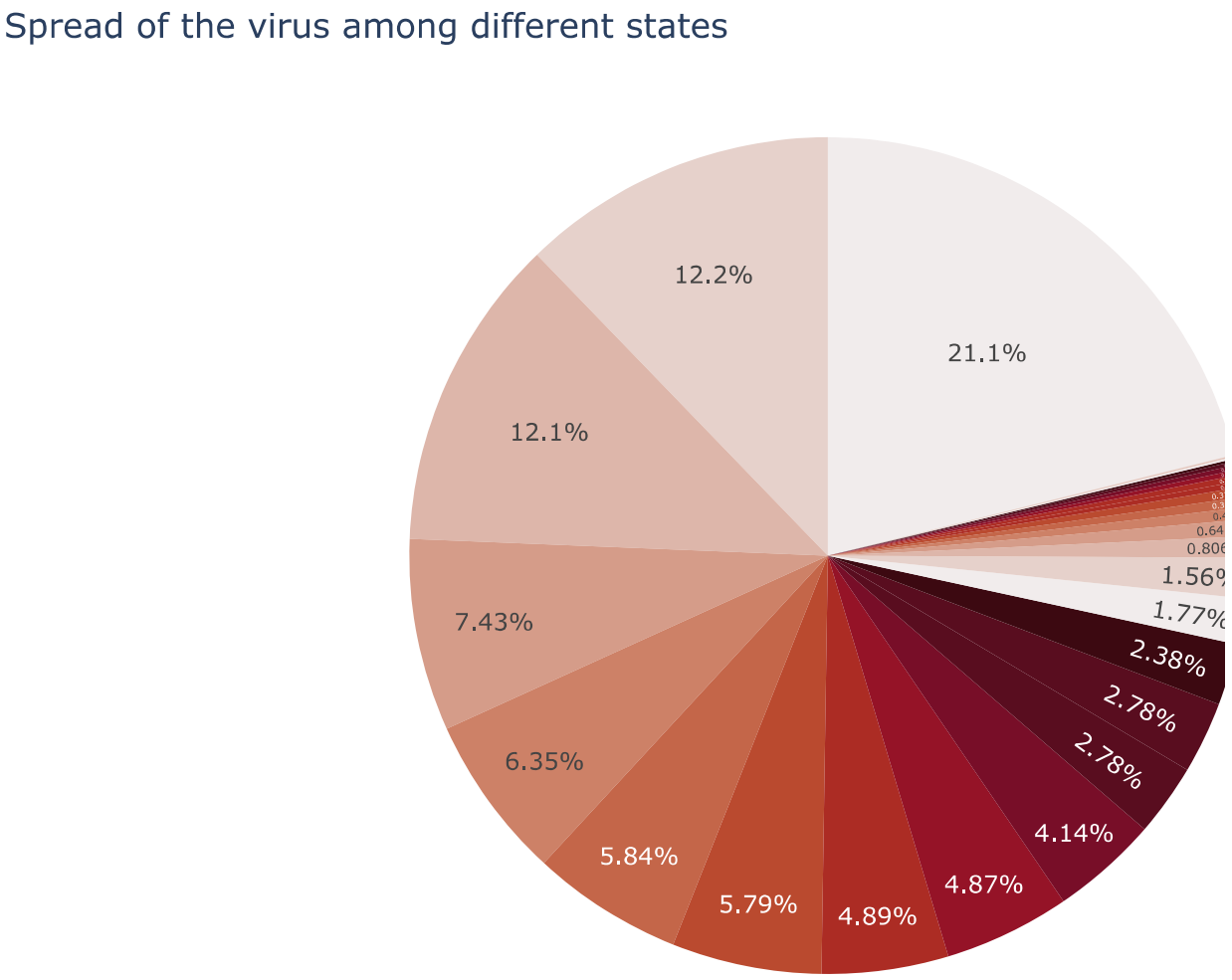
(11th April, 2020)

```
# latest data
latest = df[df['Date']==max(df['Date'])]

# days
latest_day = max(df['Date'])
day_before = latest_day - timedelta(days = 1)

latest_day_df = df[df['Date']==latest_day].set_index('State/UT')
day_before_df = df[df['Date']==day_before].set_index('State/UT')

# merge data
```

```
temp = pd.merge(left = latest_day_df, right = day_before_df, on='State/UT', suffixes=('_lat',
# new cases
latest_day_df['New cases'] = temp['Confirmed_lat'] - temp['Confirmed_bfr']

# reset index
latest = latest_day_df.reset_index()

# fill na
latest.fillna(1, inplace=True)

temp = latest[['State/UT', 'Confirmed', 'Active', 'New cases', 'Deaths', 'Mortality rate', '(
temp = temp.sort_values('Confirmed', ascending=False).reset_index(drop=True)

temp.style\
    .background_gradient(cmap="Blues", subset=['Confirmed', 'Active', 'New cases'])\
    .background_gradient(cmap="Greens", subset=['Cured', 'Recovery rate'])\
    .background_gradient(cmap="Reds", subset=['Deaths', 'Mortality rate'])
```

	State/UT	Confirmed	Active	New cases	Deaths	Mortality rate	Cured	Recovery
0	Maharashtra	1574	1276	210	110	0.069886	188	0.119441
1	Tamil Nadu	911	850	77	8	0.008782	44	0.048299

```
fig = px.pie(temp, values='Confirmed', names='State/UT', color= 'Confirmed',
             color_discrete_sequence=px.colors.sequential.amp,title="Spread of the virus amor
fig.update_traces(textposition='inside', textinfo='percent')
fig.show()
```

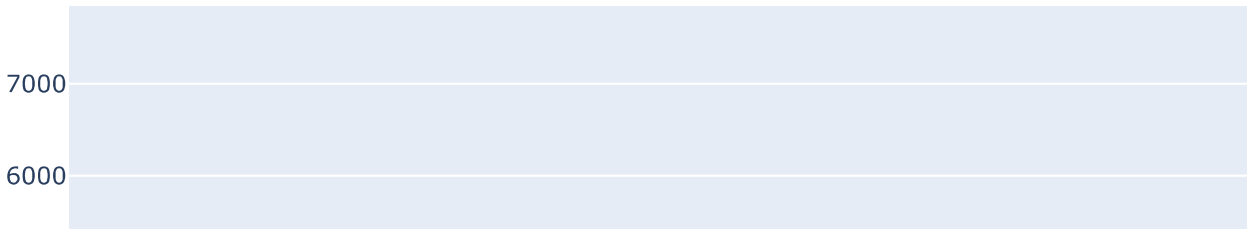



```
temp = latest.melt(id_vars="Date", value_vars=['Cured', 'Deaths', 'Active'],
                   var_name='Case', value_name='Count')
temp.head()
```

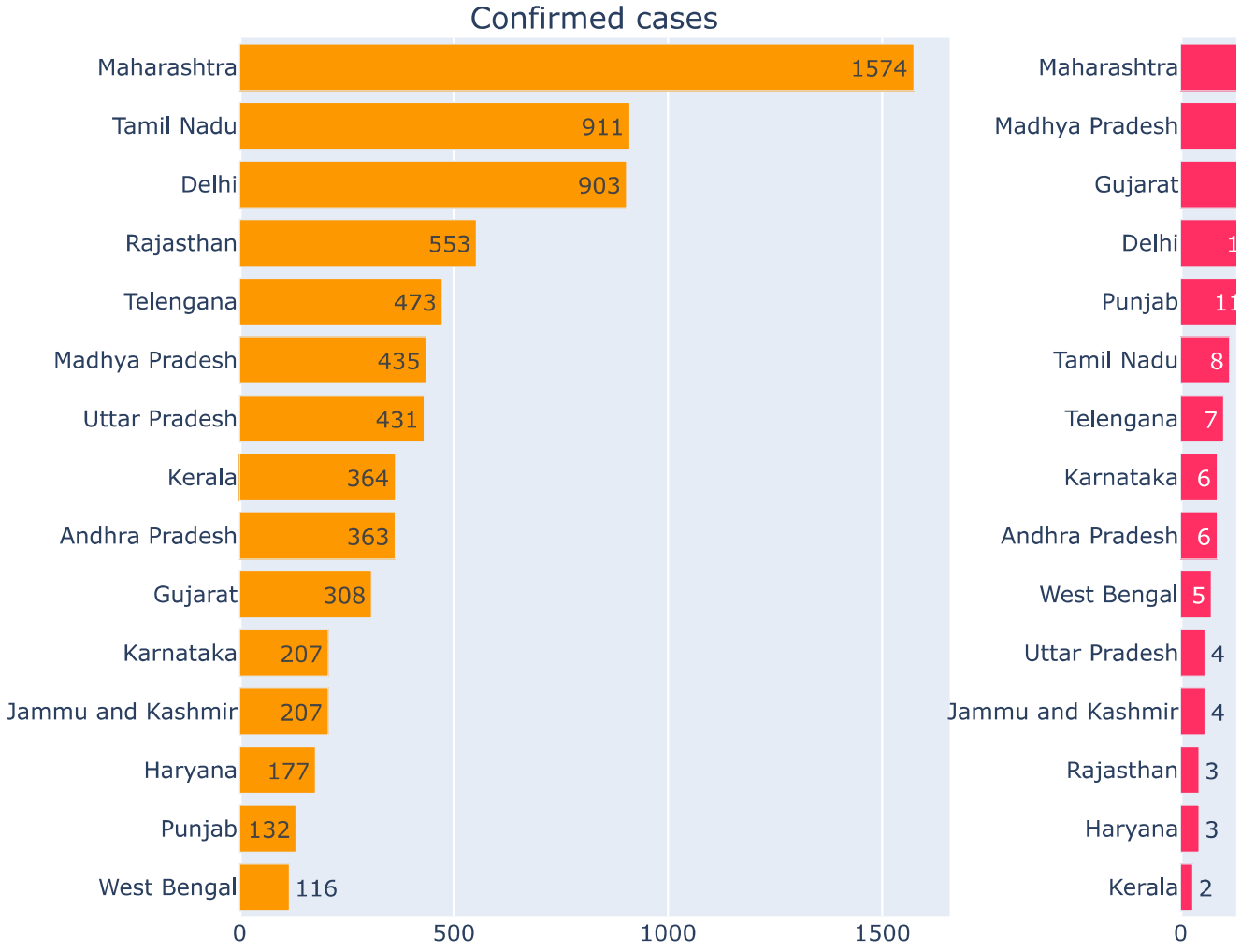
	Date	Case	Count
0	2020-04-11	Cured	0
1	2020-04-11	Cured	7

```
fig = px.treemap(temp, path=["Case"], values="Count", height=250, width=800,
                 color_discrete_sequence=[act, rec, dth], title='Latest stats of 2020-04-11')
fig.data[0].textinfo = 'label+text+value'
fig.show()
```


```
temp = df.groupby('Date')['Cured', 'Deaths', 'Active'].sum().reset_index()
temp = temp.melt(id_vars="Date", value_vars=['Cured', 'Deaths', 'Active'],
                 var_name='Case', value_name='Count')
temp.head()

fig = px.bar(temp, x="Date", y="Count", color='Case', height=540, title='Cases over time', co
fig.show()
```

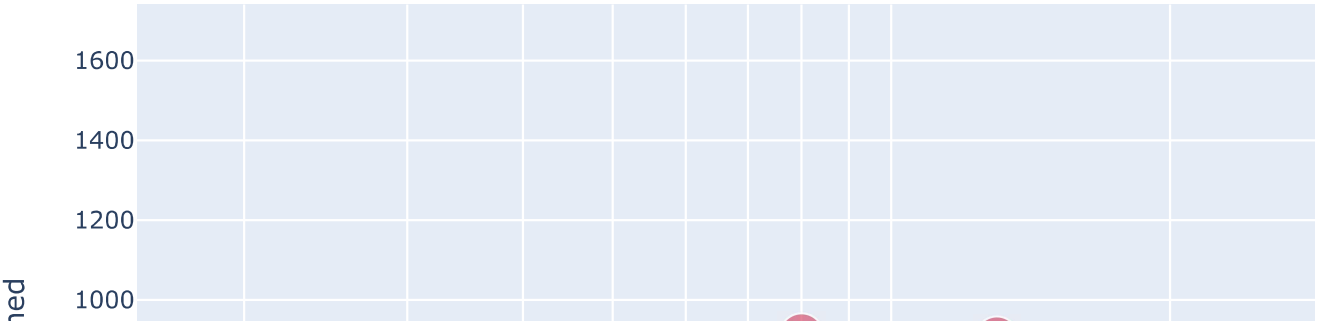



```
fig_c = px.bar(latest.sort_values('Confirmed').tail(15), x="Confirmed", y="State/UT",
               text='Confirmed', orientation='h', color_discrete_sequence = [act])
fig_d = px.bar(latest.sort_values('Deaths').tail(15), x="Deaths", y="State/UT",
               text='Deaths', orientation='h', color_discrete_sequence = [dth])
fig_r = px.bar(latest.sort_values('Cured').tail(15), x="Cured", y="State/UT",
               text='Cured', orientation='h', color_discrete_sequence = [rec])
fig_a = px.bar(latest.sort_values('Active').tail(15), x="Active", y="State/UT",
               text='Active', orientation='h', color_discrete_sequence = ['#333333'])


fig = make_subplots(rows=2, cols=2, shared_xaxes=False, horizontal_spacing=0.14, vertical_spa
                    subplot_titles=('Confirmed cases', 'Deaths reported', 'Recovered', 'Activ

fig.add_trace(fig_c['data'][0], row=1, col=1)
fig.add_trace(fig_d['data'][0], row=1, col=2)
fig.add_trace(fig_r['data'][0], row=2, col=1)
fig.add_trace(fig_a['data'][0], row=2, col=2)

fig.update_layout(height=1200)
```



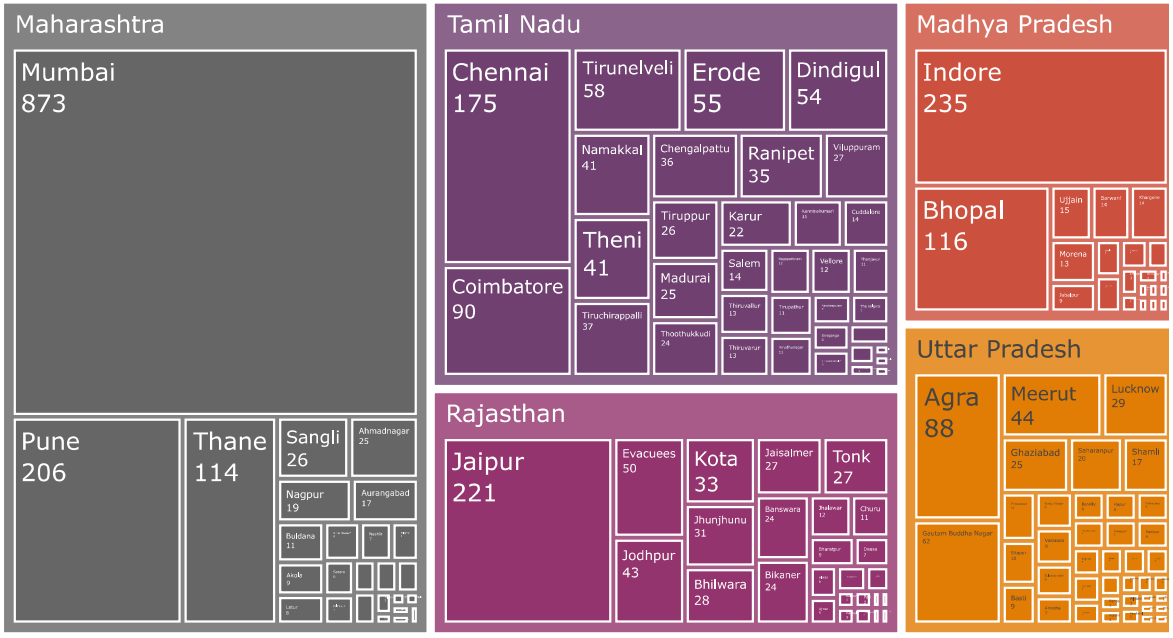

```
px.scatter(latest[latest['Deaths']>1], x='Deaths', y='Confirmed', color='Confirmed', size='Co
           title='Confirmed vs Death',hover_data=['Confirmed','State/UT','Deaths'])
```

Confirmed vs Death

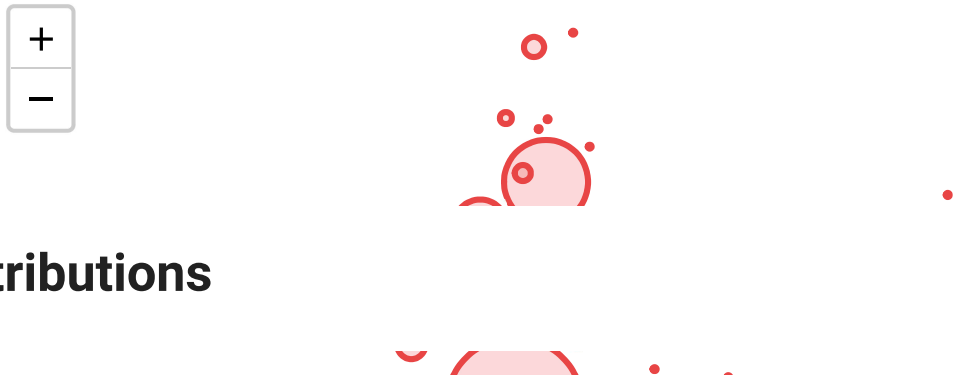



```
dist = p_df.groupby(['detected_state', 'detected_district'])['patient_number'].count().reset_
dist.head()
fig = px.treemap(dist, path=["detected_state", "detected_district"], values="patient_number",
           title='Distribution of cases around the country', color_discrete_sequence = px.col
fig.data[0].textinfo = 'label+text+value'
fig.show()
```

Distribution of cases around the country



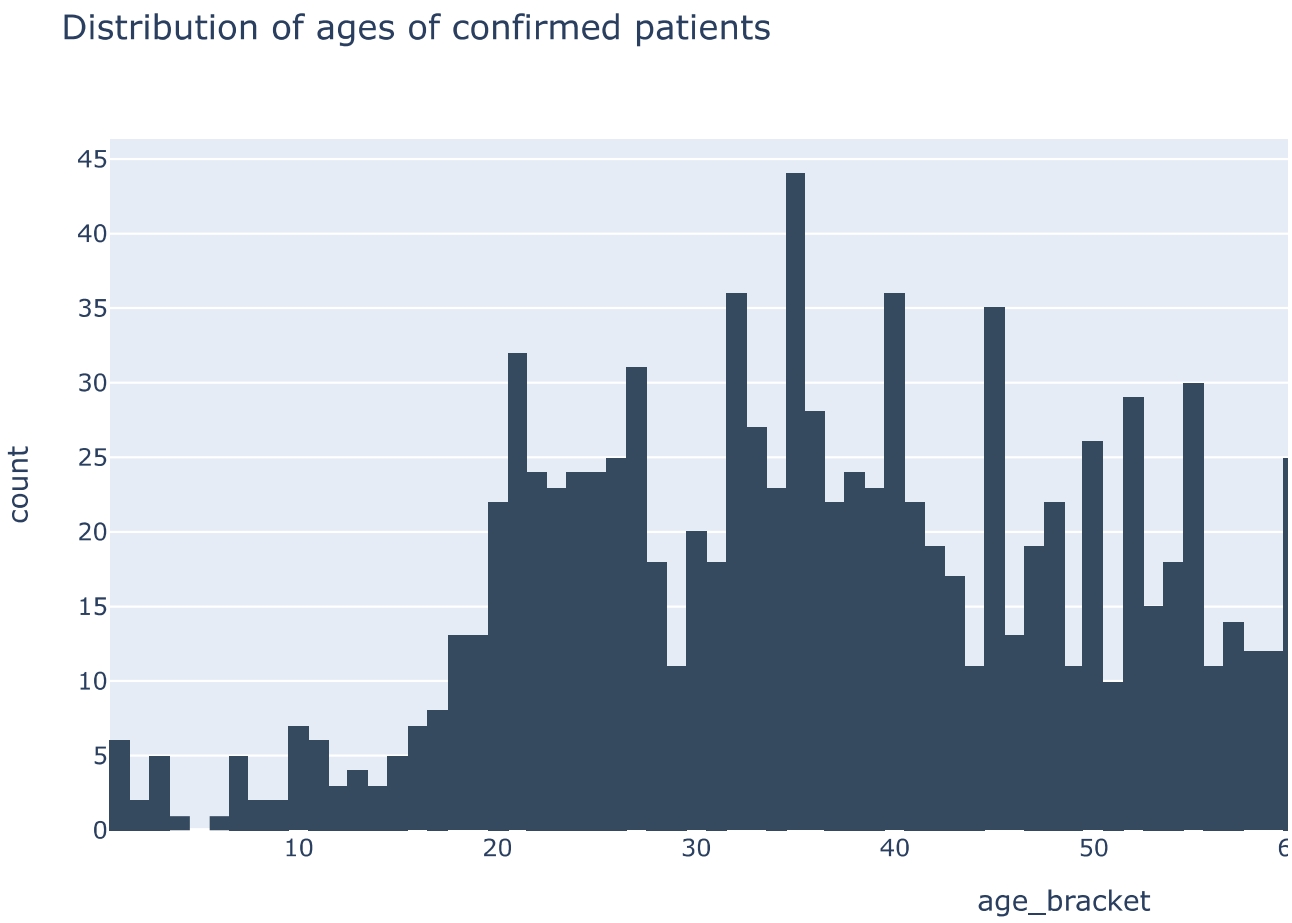
```
m = folium.Map(location = [20.5937, 78.9629], tiles = 'cartodbpositron',
               min_zoom = 3, max_zoom = 6, zoom_start = 3.5,width=600,height=400)

for i in range(0, len(latest)):
    if latest.iloc[i]['Confirmed'] > 0:
        folium.Circle(
            location = [latest.iloc[i]['Latitude'], latest.iloc[i]
                        ['Longitude']],
            color = '#e84545',
            fill = '#e84545',
            tooltip = '<li><bold>Name of Sate / UT :' + str(latest.iloc[i]
                       ['State/UT']) +
                       '<li><bold>Confirmed cases :' + str(latest.iloc[i]
                       ['Confirmed']),
            radius = int(latest.iloc[i]['Confirmed'])*200
        ).add_to(m)
m
```



Distributions

```
print('Total no. of values :', p_df.shape[0], '\nNo. of missing values :', p_df.shape[0]-p_df
px.histogram(p_df['age_bracket'], x='age_bracket', color_discrete_sequence = ['#35495f'], tit
```

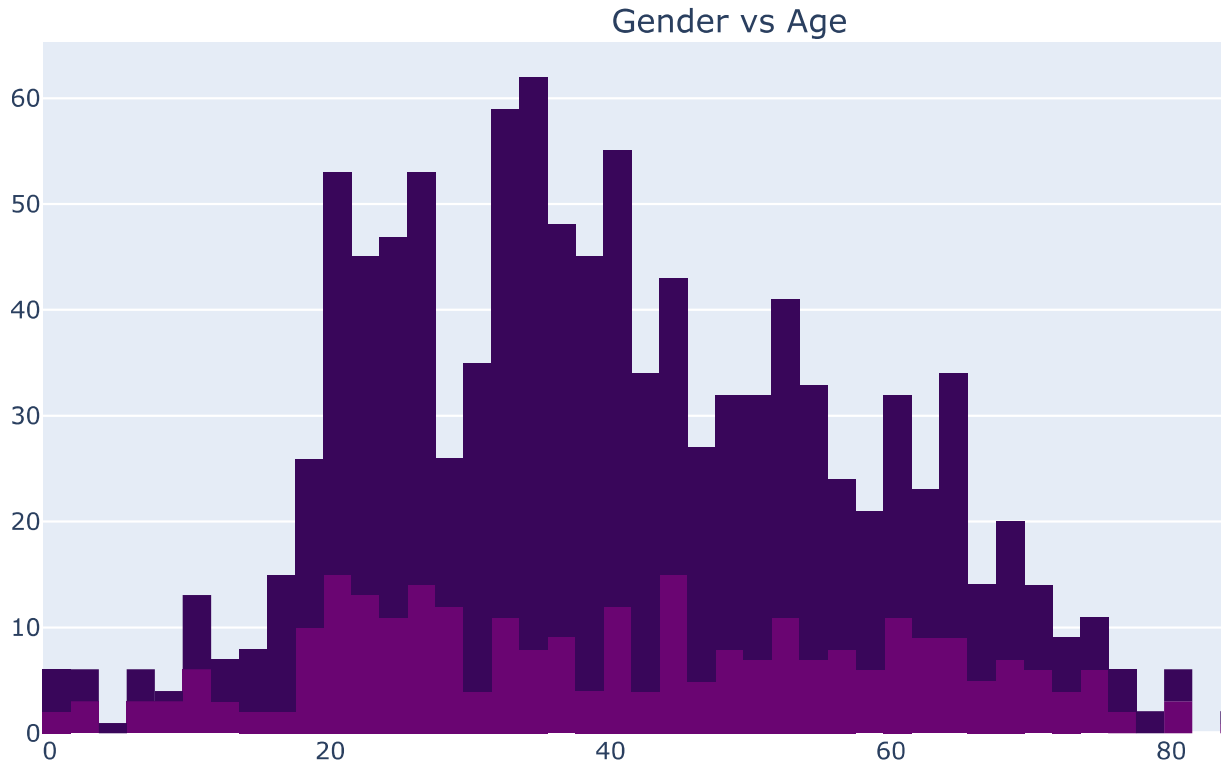
```
Total no. of values : 7618
No. of missing values : 6458
No. of available values : 1160
```




```
fig = make_subplots(
    rows=1, cols=2, column_widths=[0.8, 0.2],
    subplot_titles = ['Gender vs Age', ''],
    specs=[[{"type": "histogram"}, {"type": "pie"}]]
)
```

```
temp = p_df[['age_bracket', 'gender']].dropna()
print('Total no. of values :', p_df.shape[0], '\nNo. of missing values :', p_df.shape[0]-temp
gen_grp = temp.groupby('gender').count()

fig.add_trace(go.Histogram(x=temp[temp['gender']=='F']['age_bracket'], nbinsx=50, name='Femal
fig.add_trace(go.Histogram(x=temp[temp['gender']=='M']['age_bracket'], nbinsx=50, name='Male'

fig.add_trace(go.Pie(values=gen_grp.values.reshape(-1).tolist(), labels=['Female', 'Male'], m

fig.update_layout(showlegend=False)
fig.update_layout(barmode='stack')
fig.data[2].textinfo = 'label+text+value+percent'
fig.show()
```

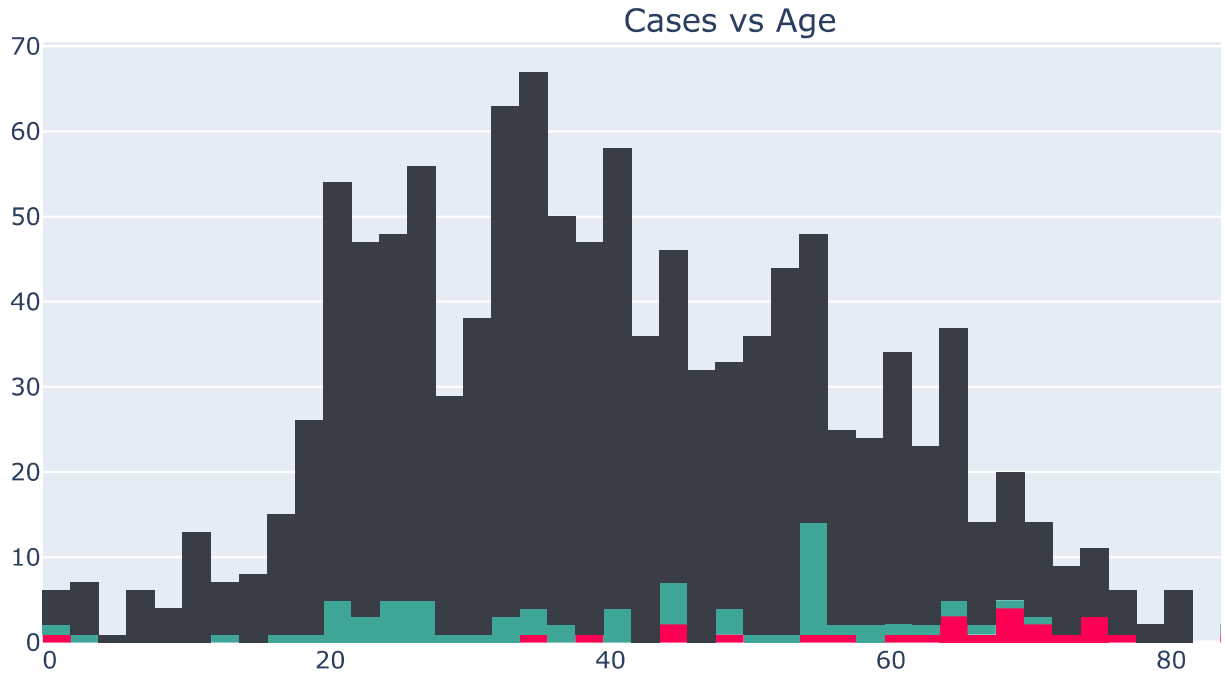
```
Total no. of values : 7618
No. of missing values : 6530
No. of available values : 1088
```




```
fig = make_subplots(
    rows=1, cols=2, column_widths=[0.8, 0.2],
    subplot_titles = ['Cases vs Age', ''],
    specs=[[{"type": "histogram"}, {"type": "pie"}]]
```

```
)

temp = p_df[['age_bracket', 'current_status']].dropna()
print('Total no. of values :', p_df.shape[0], '\nNo. of missing values :', p_df.shape[0]-temp
gen_grp = temp.groupby('current_status').count()

fig.add_trace(go.Pie(values=gen_grp.values.reshape(-1).tolist(), labels=['Deceased', 'Hospita
                     marker_colors = ['#fd0054', '#393e46', '#40a798'], hole=.3),1, 2)

fig.add_trace(go.Histogram(x=temp[temp['current_status']=='Deceased']['age_bracket'], nbinsx=
fig.add_trace(go.Histogram(x=temp[temp['current_status']=='Recovered']['age_bracket'], nbinsx
fig.add_trace(go.Histogram(x=temp[temp['current_status']=='Hospitalized']['age_bracket'], nbi

fig.update_layout(showlegend=False)
fig.update_layout(barmode='stack')
fig.data[0].textinfo = 'label+text+value+percent'

fig.show()
```

```
Total no. of values : 7618
No. of missing values : 6458
No. of available values : 1160
```



Chronology of the spread on the basis of type of the case

Imported or Local cases

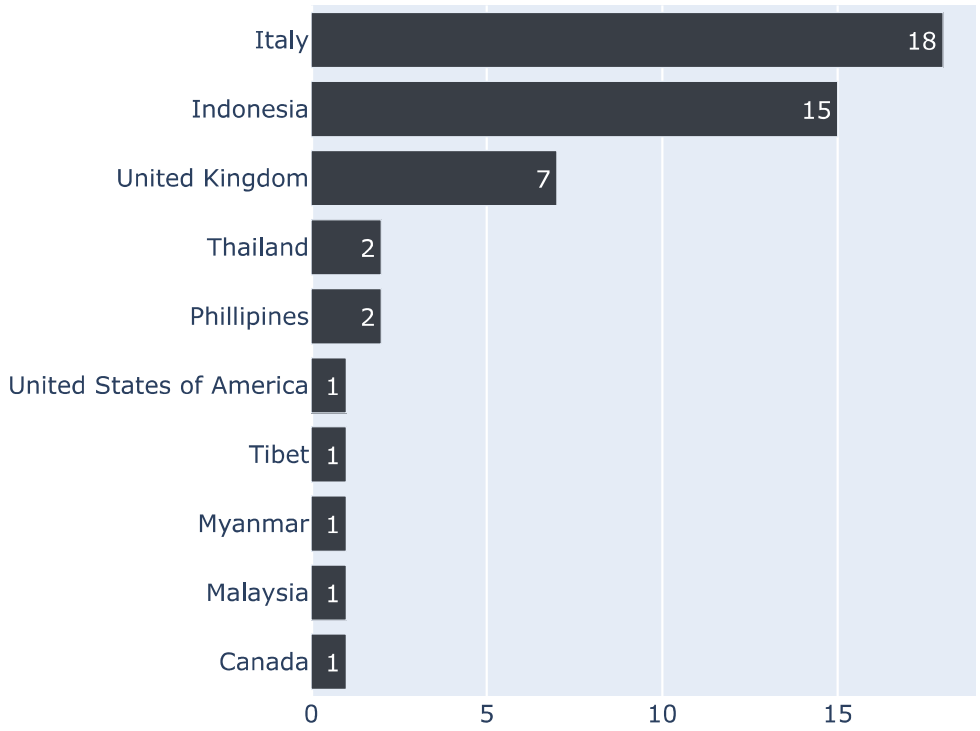
```
temp = p_df[['date_announced','type_of_transmission']].copy()
temp = temp.dropna()


temp= temp.pivot_table(index='date_announced',columns='type_of_transmission',aggfunc=len)


fig = go.Figure()

fig.add_trace(go.Scatter(y=temp['Imported'],
                         mode='lines+markers', name='Imported Cases',connectgaps=True,line=di

fig.add_trace(go.Scatter(y=temp['Local'], hoverinfo='x+y',
                         mode='lines+markers', name='Local Cases',connectgaps=True,line=dict(

fig.update_layout(title='It started with imported cases visible among the people leading to l
                  xaxis_title='Days', yaxis_title='Cases')
fig.update_traces(textposition='top center')
fig.update_layout(uniformtext_minsize=8, uniformtext_mode='hide')
fig.show()
```

It started with imported cases visible among the people leading to local

Foriegn Cases

200

```
temp = p_df.groupby('nationality')['patient_number'].count().reset_index()
temp = temp.sort_values('patient_number')
temp = temp[temp['nationality']!='India']
fig = px.bar(temp, x='patient_number', y='nationality', orientation='h', text='patient_number
       color_discrete_sequence = [cnf], title='No. of foreign citizens')
fig.update_xaxes(title='')
fig.update_yaxes(title='')
fig.show()
```

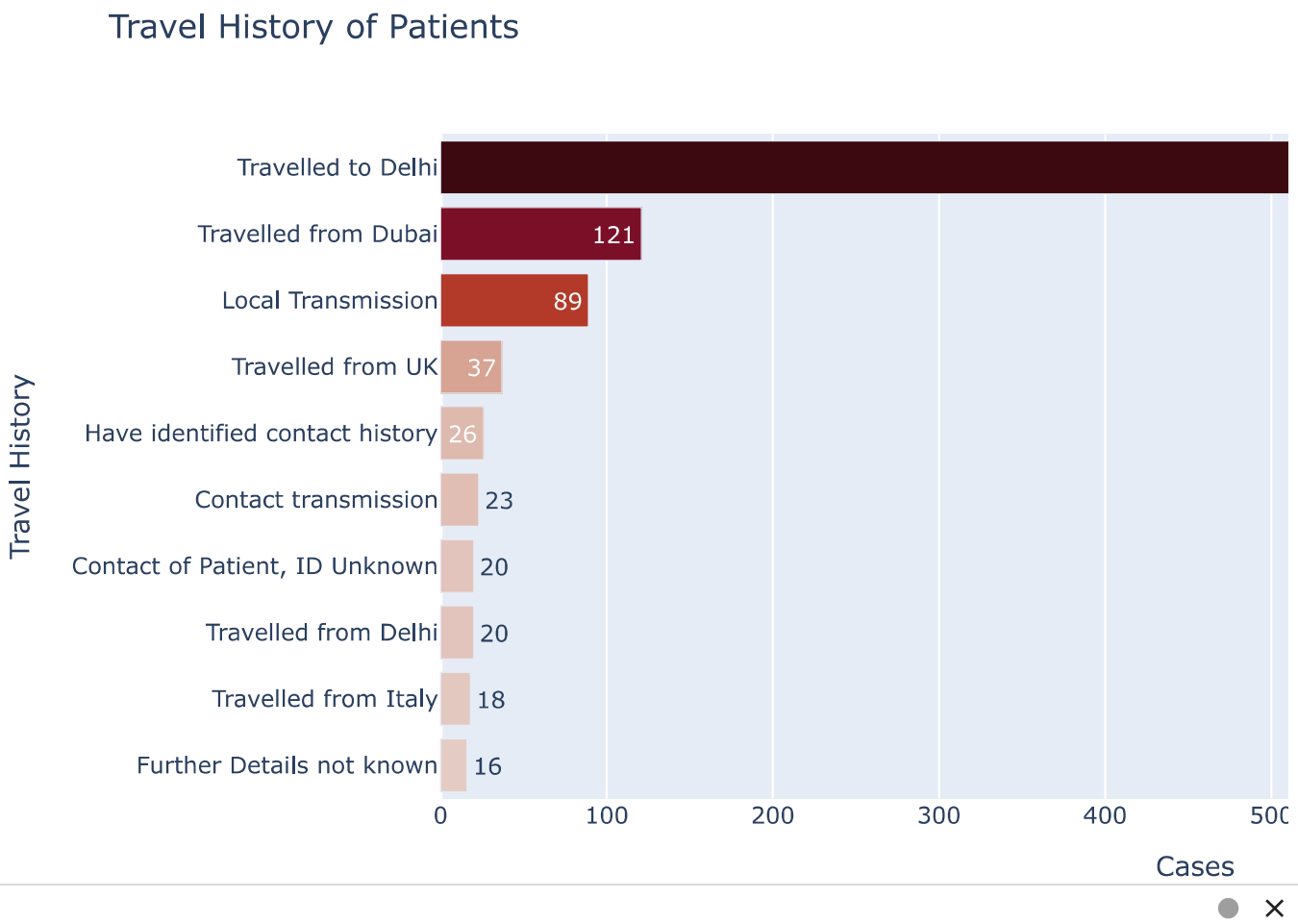


Travel history of the patients

```
p_df['notes'] = p_df['notes'].replace('Details Awaited', 'Details awaited')
p_df['notes'] = p_df['notes'].replace('Travelled from Dubai, UAE', 'Travelled from Dubai')
p_df['notes'] = p_df['notes'].replace('attended religious event Tablighi Jamaat in delhi', 'A
p_df['notes'] = p_df['notes'].replace('Travelled from London', 'Travelled from UK')
p_df['notes'] = p_df['notes'].replace('Travelled from Dubai.', 'Travelled from Dubai')
temp = pd.DataFrame(p_df.groupby('notes')['notes'].count().sort_values(ascending=False))
temp.columns = ['Count']
temp = temp.reset_index()
temp = temp[temp['notes']!='Details awaited']
temp.head(5)
```

	notes	Count
1	Travelled to Delhi	863
2	Travelled from Dubai	121
3	Local Transmission	89
4	Travelled from UK	37
5	Have identified contact history	26

```
fig = px.bar(temp.head(10).sort_values('Count', ascending=True), x='Count', y='notes', orient
             range_color=[0,150],color_continuous_scale='amp',title='Travel History of Patier
fig.update_xaxes(title='Cases')
fig.update_yaxes(title='Travel History')
```


Travel History of Patients
Travelled to Delhi
Travelled from Dubai
121
Local Transmission
89
Travelled from UK
37
Have identified contact history
26
Contact transmission
23
Contact of Patient, ID Unknown
20
Travelled from Delhi
20
Travelled from Italy
18
Further Details not known
16
Travel History
0
100
200
300
400
500
Cases