# Final Project: Covid-19

```
In [34]: import pandas as pd
         import numpy as np
         import matplotlib.pyplot as plt
         import seaborn as seabornInstance
         from sklearn.model_selection import train_test_split
         from sklearn.linear_model import LinearRegression
         from sklearn import metrics
         %matplotlib inline
         import plotly as py

         import seaborn as sns
         import datetime
```

## Import Datasets

```
In [35]: countries = pd.read_csv("4.18states.csv")
         us_counties = pd.read_csv("abridged_couties.csv")
         us_confirmed_dates = pd.read_csv("time_series_covid19_confirmed_US.csv")
         us_confirmed_deaths = pd.read_csv("time_series_covid19_deaths_US.csv")
```

In [36]:

```
countries.head()
```

Out[36]:

| | Province_State | Country_Region | Last_Update | Lat | Long_ | Confirmed | Deaths | Recovered | Active | FIPS | Incident_Rate | People_Tested |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | Alabama | US | 2020-04-18 22:32:47 | 32.3182 | -86.9023 | 4712 | 153 | NaN | 4559.0 | 1.0 | 100.492717 | 42538.0 |
| **1** | Alaska | US | 2020-04-18 22:32:47 | 61.3707 | -152.4044 | 314 | 9 | 147.0 | 305.0 | 2.0 | 52.530410 | 9655.0 |
| **2** | American Samoa | US | NaN | -14.2710 | -170.1320 | 0 | 0 | NaN | NaN | 60.0 | 0.000000 | 3.0 |
| **3** | Arizona | US | 2020-04-18 22:32:47 | 33.7298 | -111.4312 | 4724 | 180 | 539.0 | 4544.0 | 4.0 | 64.901548 | 51045.0 |
| **4** | Arkansas | US | 2020-04-18 22:32:47 | 34.9697 | -92.3731 | 1744 | 38 | 703.0 | 1706.0 | 5.0 | 67.361213 | 24141.0 |

In [37]:

```
us_counties.head(5)
```

Out[37]:

| | countyFIPS | STATEFP | COUNTYFP | CountyName | StateName | State | lat | lon | POP_LATITUDE | POP_LONGITUDE | ... | >500 gatherings |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 01001 | 1.0 | 1.0 | Autauga | AL | Alabama | 32.540091 | -86.645649 | 32.500389 | -86.494165 | ... | 737497.0 |
| **1** | 01003 | 1.0 | 3.0 | Baldwin | AL | Alabama | 30.738314 | -87.726272 | 30.548923 | -87.762381 | ... | 737497.0 |
| **2** | 01005 | 1.0 | 5.0 | Barbour | AL | Alabama | 31.874030 | -85.397327 | 31.844036 | -85.310038 | ... | 737497.0 |
| **3** | 01007 | 1.0 | 7.0 | Bibb | AL | Alabama | 32.999024 | -87.125260 | 33.030921 | -87.127659 | ... | 737497.0 |
| **4** | 01009 | 1.0 | 9.0 | Blount | AL | Alabama | 33.990440 | -86.562711 | 33.955243 | -86.591491 | ... | 737497.0 |

5 rows × 87 columns

Convert null values in `State` by examining their `StateName` and using a dictionary with key, value pairs to apply the transformation.

In [38]:

```
abbrev_to_state = {
        'AK': 'Alaska',
        'AL': 'Alabama',
        'AR': 'Arkansas',
        'AS': 'American Samoa',
        'AZ': 'Arizona',
        'CA': 'California',
        'CO': 'Colorado',
        'CT': 'Connecticut',
        'DC': 'District of Columbia',
        'DE': 'Delaware',
        'FL': 'Florida',
        'GA': 'Georgia',
        'GU': 'Guam',
        'HI': 'Hawaii',
        'IA': 'Iowa',
        'ID': 'Idaho',
        'IL': 'Illinois',
        'IN': 'Indiana',
        'KS': 'Kansas',
        'KY': 'Kentucky',
        'LA': 'Louisiana',
        'MA': 'Massachusetts',
        'MD': 'Maryland',
        'ME': 'Maine',
        'MI': 'Michigan',
        'MN': 'Minnesota',
        'MO': 'Missouri',
        'MP': 'Northern Mariana Islands',
        'MS': 'Mississippi',
        'MT': 'Montana',
        'NA': 'National',
        'NC': 'North Carolina',
        'ND': 'North Dakota',
        'NE': 'Nebraska',
        'NH': 'New Hampshire',
        'NJ': 'New Jersey',
        'NM': 'New Mexico',
        'NV': 'Nevada',
        'NY': 'New York',
        'OH': 'Ohio',
        'OK': 'Oklahoma',
        'OR': 'Oregon',
        'PA': 'Pennsylvania',
        'PR': 'Puerto Rico',
        'RI': 'Rhode Island',
        'SC': 'South Carolina',
        'SD': 'South Dakota',
        'TN': 'Tennessee',
        'TX': 'Texas',
        'UT': 'Utah',
        'VA': 'Virginia',
        'VI': 'Virgin Islands',
        'VT': 'Vermont',
        'WA': 'Washington',
        'WI': 'Wisconsin',
        'WV': 'West Virginia',
        'WY': 'Wyoming'
}
us_counties['State']=us_counties['StateName'].map(abbrev_to_state)
```

In [39]:
```
us_confirmed_dates.head()
```

Out[39]:

| | UID | iso2 | iso3 | code3 | FIPS | Admin2 | Province_State | Country_Region | Lat | Long_ | ... | 4/9/20 | 4/10/20 | 4/11/20 | 4/12/20 | 4/13/20 | 4/1 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 16 | AS | ASM | 16 | 60.0 | NaN | American Samoa | US | -14.2710 | -170.1320 | ... | 0 | 0 | 0 | 0 | 0 | |
| **1** | 316 | GU | GUM | 316 | 66.0 | NaN | Guam | US | 13.4443 | 144.7937 | ... | 128 | 130 | 133 | 133 | 133 | |
| **2** | 580 | MP | MNP | 580 | 69.0 | NaN | Northern Mariana Islands | US | 15.0979 | 145.6739 | ... | 11 | 11 | 11 | 11 | 11 | |
| **3** | 630 | PR | PRI | 630 | 72.0 | NaN | Puerto Rico | US | 18.2208 | -66.5901 | ... | 683 | 725 | 788 | 897 | 903 | |
| **4** | 850 | VI | VIR | 850 | 78.0 | NaN | Virgin Islands | US | 18.3358 | -64.8963 | ... | 45 | 50 | 51 | 51 | 51 | |

5 rows × 99 columns

```
In [40]: us_confirmed_deaths.head()
```

```
Out[40]:
     UID  iso2  iso3  code3  FIPS  Admin2      Province_State  Country_Region       Lat     Long_  ...  4/9/20  4/10/20  4/11/20  4/12/20  4/13/20  4/1
  0   16    AS   ASM     16  60.0     NaN            American              US  -14.2710 -170.1320  ...       0        0        0        0        0
                                                       Samoa
  1  316    GU   GUM    316  66.0     NaN                Guam              US   13.4443  144.7937  ...       4        4        5        5        5
  2  580    MP   MNP    580  69.0     NaN            Northern              US   15.0979  145.6739  ...       2        2        2        2        2
                                                Mariana Islands
  3  630    PR   PRI    630  72.0     NaN         Puerto Rico              US   18.2208  -66.5901  ...      33       39       42       44       45
  4  850    VI   VIR    850  78.0     NaN      Virgin Islands              US   18.3358  -64.8963  ...       1        1        1        1        1

5 rows × 100 columns
```

## Objective

What factors affect the transmission rate of COVID-19? How can we predict recovery rates based on these factors across the U.S.?

Train a machine learning model to predict `Recovery_Rate` using **Linear Regression**.

## Data Cleaning

In [41]:

```
drop_us_county_columns = ['PopMale<52010',
                'PopFmle<52010',
                 'PopMale5-92010',
                 'PopFmle5-92010',
                 'PopMale10-142010',
                 'PopFmle10-142010',
                 'PopMale15-192010',
                 'PopFmle15-192010',
                 'PopMale20-242010',
                 'PopFmle20-242010',
                 'PopMale25-292010',
                 'PopFmle25-292010',
                 'PopMale30-342010',
                 'PopFmle30-342010',
                 'PopMale35-442010',
                 'PopFmle35-442010',
                 'PopMale45-542010',
                 'PopFmle45-542010',
                 'PopMale55-592010',
                 'PopFmle55-592010',
                 'PopMale60-642010',
                 'PopFmle60-642010',
                 'PopMale65-742010',
                 'PopFmle65-742010',
                 'PopMale75-842010',
                 'PopFmle75-842010',
                 'PopMale>842010',
                 'PopFmle>842010',
                'countyFIPS',
                'StateName',
                'STATEFP',
                'COUNTYFP',
                'CensusRegionName',
                'CensusDivisionName',
                'Rural-UrbanContinuumCode2013',
                 'PopTotalMale2017', 'PopTotalFemale2017', 'FracMale2017',
               'dem_to_rep_ratio',
                '3-YrMortalityAge<1Year2015-17',
                 '3-YrMortalityAge1-4Years2015-17',
                 '3-YrMortalityAge5-14Years2015-17',
                 '3-YrMortalityAge15-24Years2015-17',
                 '3-YrMortalityAge25-34Years2015-17',
                 '3-YrMortalityAge35-44Years2015-17',
                 '3-YrMortalityAge45-54Years2015-17',
                 '3-YrMortalityAge55-64Years2015-17',
                 '3-YrMortalityAge65-74Years2015-17',
                 '3-YrMortalityAge75-84Years2015-17',
                 '3-YrMortalityAge85+Years2015-17',
                 '#EligibleforMedicare2018',
                 'MedicareEnrollment,AgedTot2017']
us_counties = us_counties.drop(columns = drop_us_county_columns)
```

Convert ordinal dates in columns `stay at home` , `>50 gatherings` , `>500 gatherings` , `public schools` , `restaurant dine-in` , `entertainment/gym` , `federal guidelines` , `foreign travel ban` into timestamps.

In [42]:

```python
us_counties.iloc[:, 23:31] = us_counties.iloc[:, 23:31].fillna(0).astype(int)

def convert_timestamps(col):
    for i in range(len(us_counties[col])):
        if (us_counties[col][i] == 0):
            us_counties[col][i] = 0
        else:
            us_counties[col][i] = pd.Timestamp.fromordinal(us_counties[col][i]).date()

convert_timestamps('stay at home')
convert_timestamps('>50 gatherings')
convert_timestamps('>500 gatherings')
convert_timestamps('public schools')
convert_timestamps('restaurant dine-in')
convert_timestamps('entertainment/gym')
convert_timestamps('federal guidelines')
convert_timestamps('foreign travel ban')
```

```
/srv/conda/envs/data100/lib/python3.7/site-packages/ipykernel_launcher.py:8: SettingWithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#r
eturning-a-view-versus-a-copy

/srv/conda/envs/data100/lib/python3.7/site-packages/ipykernel_launcher.py:6: SettingWithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#r
eturning-a-view-versus-a-copy
```

In [43]:
```
guidelines = us_counties.iloc[:, 23:31]
guidelines.head(5)
```

Out[43]:

| | stay at home | >50 gatherings | >500 gatherings | public schools | restaurant dine-in | entertainment/gym | federal guidelines | foreign travel ban |
|---|---|---|---|---|---|---|---|---|
| **0** | 2020-04-04 | 2020-03-20 | 2020-03-13 | 2020-03-16 | 2020-03-19 | 2020-03-28 | 2020-03-16 | 2020-03-11 |
| **1** | 2020-04-04 | 2020-03-20 | 2020-03-13 | 2020-03-16 | 2020-03-19 | 2020-03-28 | 2020-03-16 | 2020-03-11 |
| **2** | 2020-04-04 | 2020-03-20 | 2020-03-13 | 2020-03-16 | 2020-03-19 | 2020-03-28 | 2020-03-16 | 2020-03-11 |
| **3** | 2020-04-04 | 2020-03-20 | 2020-03-13 | 2020-03-16 | 2020-03-19 | 2020-03-28 | 2020-03-16 | 2020-03-11 |
| **4** | 2020-04-04 | 2020-03-20 | 2020-03-13 | 2020-03-16 | 2020-03-19 | 2020-03-28 | 2020-03-16 | 2020-03-11 |

In [44]:
```
grouped_states = us_counties.groupby(['State']).mean()
grouped_states.head()
```

Out[44]:

| | lat | lon | POP_LATITUDE | POP_LONGITUDE | PopulationEstimate2018 | PopulationEstimate65+2017 | PopulationDensityperSqMil |
|---|---|---|---|---|---|---|---|
| **State** | | | | | | | |
| **Alabama** | 32.887935 | -86.709300 | 32.878325 | -86.700935 | 72953.298507 | 11996.582090 | 90.2 |
| **Alaska** | NaN | NaN | 60.190155 | -148.255559 | 25428.896552 | 2847.586207 | 7.7 |
| **American Samoa** | NaN | NaN | NaN | NaN | NaN | NaN | |
| **Arizona** | 33.678351 | -111.467022 | 33.581097 | -111.477913 | 478109.733333 | 80116.400000 | 52.0 |
| **Arkansas** | 34.911163 | -92.437589 | 34.924027 | -92.428029 | 40184.333333 | 6655.253333 | 54.3 |

5 rows × 25 columns

In [45]:
```
countries = countries.drop(columns=['Last_Update','FIPS', 'UID', 'ISO3'])
countries.head(5)
```

Out[45]:

| | Province_State | Country_Region | Lat | Long_ | Confirmed | Deaths | Recovered | Active | Incident_Rate | People_Tested | People_Hospitalized |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | Alabama | US | 32.3182 | -86.9023 | 4712 | 153 | NaN | 4559.0 | 100.492717 | 42538.0 | 620.0 |
| **1** | Alaska | US | 61.3707 | -152.4044 | 314 | 9 | 147.0 | 305.0 | 52.530410 | 9655.0 | 39.0 |
| **2** | American Samoa | US | -14.2710 | -170.1320 | 0 | 0 | NaN | NaN | 0.000000 | 3.0 | NaN |
| **3** | Arizona | US | 33.7298 | -111.4312 | 4724 | 180 | 539.0 | 4544.0 | 64.901548 | 51045.0 | 566.0 |
| **4** | Arkansas | US | 34.9697 | -92.3731 | 1744 | 38 | 703.0 | 1706.0 | 67.361213 | 24141.0 | 291.0 |

In [46]:
```
grouped_countries = countries.groupby(['Province_State']).mean()
grouped_countries.head()
```

Out[46]:

| | Lat | Long_ | Confirmed | Deaths | Recovered | Active | Incident_Rate | People_Tested | People_Hospitalized | Mortality_Rate | Testi |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Province_State** | | | | | | | | | | | |
| **Alabama** | 32.3182 | -86.9023 | 4712 | 153 | NaN | 4559.0 | 100.492717 | 42538.0 | 620.0 | 3.247029 | 907 |
| **Alaska** | 61.3707 | -152.4044 | 314 | 9 | 147.0 | 305.0 | 52.530410 | 9655.0 | 39.0 | 2.866242 | 1615 |
| **Alberta** | 53.9333 | -116.5765 | 2562 | 51 | 0.0 | 2511.0 | 58.053824 | NaN | NaN | 1.990632 | |
| **American Samoa** | -14.2710 | -170.1320 | 0 | 0 | NaN | NaN | 0.000000 | 3.0 | NaN | NaN | 5 |
| **Anguilla** | 18.2206 | -63.0686 | 3 | 0 | 1.0 | 2.0 | 19.997334 | NaN | NaN | 0.000000 | |

In [47]:
```
us_confirmed_dates.head()
```

Out[47]:

| | UID | iso2 | iso3 | code3 | FIPS | Admin2 | Province_State | Country_Region | Lat | Long_ | ... | 4/9/20 | 4/10/20 | 4/11/20 | 4/12/20 | 4/13/20 | 4/1 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 16 | AS | ASM | 16 | 60.0 | NaN | American Samoa | US | -14.2710 | -170.1320 | ... | 0 | 0 | 0 | 0 | 0 | |
| **1** | 316 | GU | GUM | 316 | 66.0 | NaN | Guam | US | 13.4443 | 144.7937 | ... | 128 | 130 | 133 | 133 | 133 | |
| **2** | 580 | MP | MNP | 580 | 69.0 | NaN | Northern Mariana Islands | US | 15.0979 | 145.6739 | ... | 11 | 11 | 11 | 11 | 11 | |
| **3** | 630 | PR | PRI | 630 | 72.0 | NaN | Puerto Rico | US | 18.2208 | -66.5901 | ... | 683 | 725 | 788 | 897 | 903 | |
| **4** | 850 | VI | VIR | 850 | 78.0 | NaN | Virgin Islands | US | 18.3358 | -64.8963 | ... | 45 | 50 | 51 | 51 | 51 | |

5 rows × 99 columns

In [48]:
```
us_confirmed_dates = us_confirmed_dates.drop(columns=['UID', 'iso2', 'iso3', 'code3', 'Admin2', 'FIPS'])
```

In [49]:

```
us_confirmed_dates_groupedby_states = us_confirmed_dates.groupby(['Province_State']).sum().drop(columns=['Lat'
, 'Long_'])
us_confirmed_dates_groupedby_states = us_confirmed_dates_groupedby_states.reset_index()
us_confirmed_dates_groupedby_states['Province_State'].value_counts()
us_confirmed_dates_groupedby_states = us_confirmed_dates_groupedby_states.set_index('Province_State')
us_confirmed_dates_groupedby_states = us_confirmed_dates_groupedby_states.drop(index=['American Samoa', 'Guam'
, 'Northern Mariana Islands',
                                                                      'Puerto Rico', 'Virgin Islands', 'District of Columbia',
                                                                      'Diamond Princess', 'Grand Princess'])
us_confirmed_dates_groupedby_states.head()
```

Out[49]:

| | 1/22/20 | 1/23/20 | 1/24/20 | 1/25/20 | 1/26/20 | 1/27/20 | 1/28/20 | 1/29/20 | 1/30/20 | 1/31/20 | ... | 4/9/20 | 4/10/20 | 4/11/20 | 4/12/20 | 4/13/ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Province_State** | | | | | | | | | | | | | | | | |
| **Alabama** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ... | 2703 | 2947 | 3217 | 3563 | 37 |
| **Alaska** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ... | 235 | 246 | 257 | 272 | 2 |
| **Arizona** | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | ... | 3018 | 3112 | 3393 | 3542 | 37 |
| **Arkansas** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ... | 1119 | 1171 | 1228 | 1280 | 14 |
| **California** | 0 | 0 | 0 | 0 | 2 | 2 | 2 | 2 | 2 | 3 | ... | 19710 | 21081 | 21706 | 22795 | 239 |

5 rows × 88 columns

In [50]:

```
transposed_dates1 = us_confirmed_dates_groupedby_states.T
transposed_dates1.head()
```

Out[50]:

| Province_State | Alabama | Alaska | Arizona | Arkansas | California | Colorado | Connecticut | Delaware | Florida | Georgia | ... | South Dakota | Tennessee | Texas |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **1/22/20** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ... | 0 | 0 | 0 |
| **1/23/20** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ... | 0 | 0 | 0 |
| **1/24/20** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ... | 0 | 0 | 0 |
| **1/25/20** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ... | 0 | 0 | 0 |
| **1/26/20** | 0 | 0 | 1 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | ... | 0 | 0 | 0 |

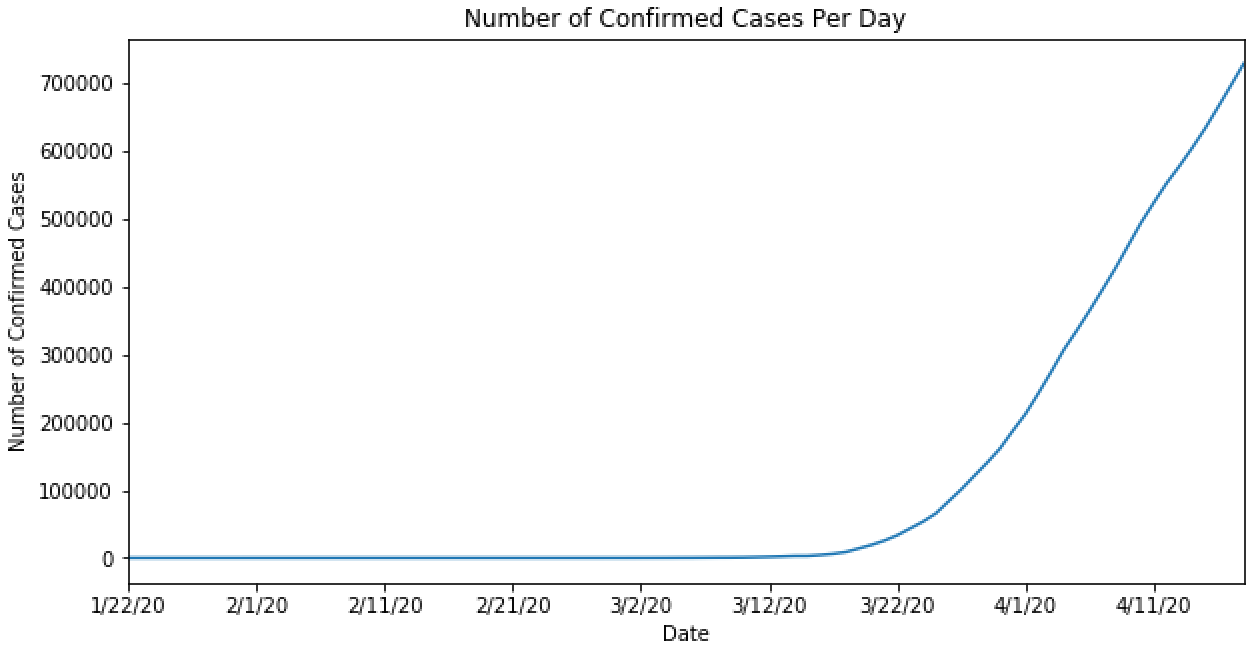
5 rows × 50 columns

In [51]:

```
transposed_dates1['Total per Day'] = transposed_dates1.sum(axis=1)
transposed_dates1['Total per Day'].plot(figsize=(10, 5))
plt.xlabel('Date')
plt.ylabel('Number of Confirmed Cases')
plt.title('Number of Confirmed Cases Per Day')
```

Out[51]:

```
Text(0.5, 1.0, 'Number of Confirmed Cases Per Day')
```



In [52]:

```
us_confirmed_deaths = us_confirmed_deaths.drop(columns=['UID', 'iso2', 'iso3', 'code3', 'FIPS', 'Admin2', 'Com
bined_Key',
                                                       'Country_Region', 'Lat', 'Long_', 'Population'])
us_confirmed_deaths.head()
```

Out[52]:

| | Province_State | 1/22/20 | 1/23/20 | 1/24/20 | 1/25/20 | 1/26/20 | 1/27/20 | 1/28/20 | 1/29/20 | 1/30/20 | ... | 4/9/20 | 4/10/20 | 4/11/20 | 4/12/20 | 4/13/20 | 4/ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | American Samoa | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ... | 0 | 0 | 0 | 0 | 0 | |
| **1** | Guam | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ... | 4 | 4 | 5 | 5 | 5 | |
| **2** | Northern Mariana Islands | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ... | 2 | 2 | 2 | 2 | 2 | |
| **3** | Puerto Rico | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ... | 33 | 39 | 42 | 44 | 45 | |
| **4** | Virgin Islands | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ... | 1 | 1 | 1 | 1 | 1 | |

5 rows × 89 columns

In [53]:

```
us_confirmed_deaths = us_confirmed_deaths.set_index('Province_State')
us_confirmed_deaths= us_confirmed_deaths.drop(index=['American Samoa', 'Guam', 'Northern Mariana Islands',
                                                    'Puerto Rico', 'Virgin Islands', 'District of Columbia',
                                                    'Diamond Princess', 'Grand Princess'])
```

```
In [54]: us_confirmed_deaths_groupedby_state = us_confirmed_deaths.groupby('Province_State').sum()
         transposed_dates2 = us_confirmed_deaths_groupedby_state.T
         transposed_dates2.head(5)
```

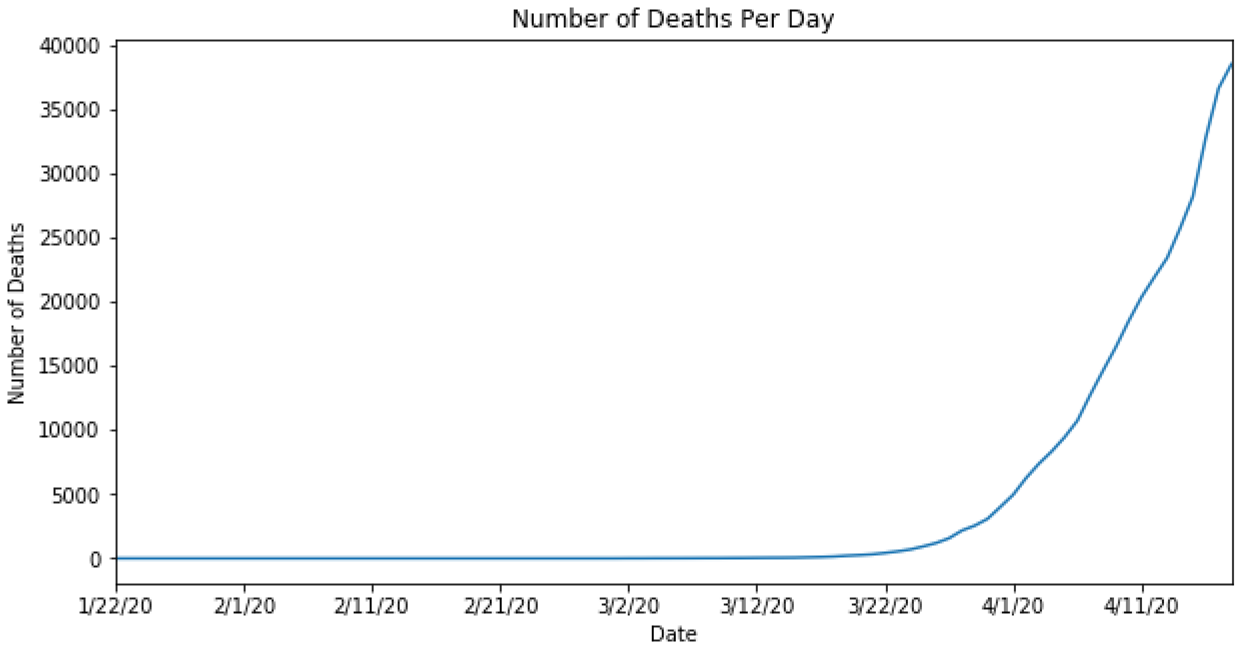
Out[54]:

| Province_State | Alabama | Alaska | Arizona | Arkansas | California | Colorado | Connecticut | Delaware | Florida | Georgia | ... | South Dakota | Tennessee | Texas |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **1/22/20** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ... | 0 | 0 | 0 |
| **1/23/20** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ... | 0 | 0 | 0 |
| **1/24/20** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ... | 0 | 0 | 0 |
| **1/25/20** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ... | 0 | 0 | 0 |
| **1/26/20** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ... | 0 | 0 | 0 |

5 rows × 50 columns

```
In [55]: transposed_dates2['Total per Day'] = transposed_dates2.sum(axis=1)
         transposed_dates2['Total per Day'].plot(figsize=(10,5))
         plt.xlabel('Date')
         plt.ylabel('Number of Deaths')
         plt.title('Number of Deaths Per Day')
```

```
Out[55]: Text(0.5, 1.0, 'Number of Deaths Per Day')
```



## GROUPED TABLES

In [56]:
```
grouped_states.head(5)
```

Out[56]:

| | lat | lon | POP_LATITUDE | POP_LONGITUDE | PopulationEstimate2018 | PopulationEstimate65+2017 | PopulationDensityperSqMil |
|---|---|---|---|---|---|---|---|
| **State** | | | | | | | |
| **Alabama** | 32.887935 | -86.709300 | 32.878325 | -86.700935 | 72953.298507 | 11996.582090 | 90.2 |
| **Alaska** | NaN | NaN | 60.190155 | -148.255559 | 25428.896552 | 2847.586207 | 7.7 |
| **American Samoa** | NaN | NaN | NaN | NaN | NaN | NaN | |
| **Arizona** | 33.678351 | -111.467022 | 33.581097 | -111.477913 | 478109.733333 | 80116.400000 | 52.0 |
| **Arkansas** | 34.911163 | -92.437589 | 34.924027 | -92.428029 | 40184.333333 | 6655.253333 | 54.3 |

5 rows × 25 columns

In [57]:
```
grouped_countries.head(5)
```

Out[57]:

| | Lat | Long_ | Confirmed | Deaths | Recovered | Active | Incident_Rate | People_Tested | People_Hospitalized | Mortality_Rate | Testi |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **Province_State** | | | | | | | | | | | |
| **Alabama** | 32.3182 | -86.9023 | 4712 | 153 | NaN | 4559.0 | 100.492717 | 42538.0 | 620.0 | 3.247029 | 907 |
| **Alaska** | 61.3707 | -152.4044 | 314 | 9 | 147.0 | 305.0 | 52.530410 | 9655.0 | 39.0 | 2.866242 | 1615 |
| **Alberta** | 53.9333 | -116.5765 | 2562 | 51 | 0.0 | 2511.0 | 58.053824 | NaN | NaN | 1.990632 | |
| **American Samoa** | -14.2710 | -170.1320 | 0 | 0 | NaN | NaN | 0.000000 | 3.0 | NaN | NaN | 5 |
| **Anguilla** | 18.2206 | -63.0686 | 3 | 0 | 1.0 | 2.0 | 19.997334 | NaN | NaN | 0.000000 | |

In [58]:

```
us_confirmed_dates_groupedby_states.head(5)
```

Out[58]:

| | 1/22/20 | 1/23/20 | 1/24/20 | 1/25/20 | 1/26/20 | 1/27/20 | 1/28/20 | 1/29/20 | 1/30/20 | 1/31/20 | ... | 4/9/20 | 4/10/20 | 4/11/20 | 4/12/20 | 4/13/ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Province_State** | | | | | | | | | | | | | | | | |
| **Alabama** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ... | 2703 | 2947 | 3217 | 3563 | 37 |
| **Alaska** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ... | 235 | 246 | 257 | 272 | 2 |
| **Arizona** | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | ... | 3018 | 3112 | 3393 | 3542 | 37 |
| **Arkansas** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ... | 1119 | 1171 | 1228 | 1280 | 14 |
| **California** | 0 | 0 | 0 | 0 | 2 | 2 | 2 | 2 | 2 | 3 | ... | 19710 | 21081 | 21706 | 22795 | 239 |

5 rows × 88 columns

In [59]:

```
us_confirmed_deaths_groupedby_state.head(5)
```

Out[59]:

| | 1/22/20 | 1/23/20 | 1/24/20 | 1/25/20 | 1/26/20 | 1/27/20 | 1/28/20 | 1/29/20 | 1/30/20 | 1/31/20 | ... | 4/9/20 | 4/10/20 | 4/11/20 | 4/12/20 | 4/13/ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Province_State** | | | | | | | | | | | | | | | | |
| **Alabama** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ... | 70 | 80 | 92 | 93 | |
| **Alaska** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ... | 7 | 7 | 8 | 8 | |
| **Arizona** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ... | 89 | 97 | 108 | 115 | 1 |
| **Arkansas** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ... | 21 | 21 | 25 | 27 | |
| **California** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | ... | 544 | 583 | 604 | 640 | 7 |

5 rows × 88 columns

## BAR PLOTS BY REGIONS

In [60]:
```
join_countries_and_states = grouped_states.join(grouped_countries)
join_countries_and_states = join_countries_and_states.drop(index = ['American Samoa', 'District of Columbia',
                                                                   'Guam', 'Northern Mariana Islands',
                                                                   'Puerto Rico', 'Virgin Islands'])
```

In [61]:
```
join_countries_and_states['Recovery_Rate'] = join_countries_and_states['Recovered'] / join_countries_and_states['Confirmed']
join_countries_and_states.head(5)
join_countries_and_states['Recovery_Rate']=join_countries_and_states['Recovery_Rate'].fillna(np.mean(join_countries_and_states['Recovery_Rate']))
join_countries_and_states.head()
```

Out[61]:

| | lat | lon | POP_LATITUDE | POP_LONGITUDE | PopulationEstimate2018 | PopulationEstimate65+2017 | PopulationDensityperSqMil |
|---|---|---|---|---|---|---|---|
| **State** | | | | | | | |
| **Alabama** | 32.887935 | -86.709300 | 32.878325 | -86.700935 | 72953.298507 | 11996.582090 | 90.2 |
| **Alaska** | NaN | NaN | 60.190155 | -148.255559 | 25428.896552 | 2847.586207 | 7.7 |
| **Arizona** | 33.678351 | -111.467022 | 33.581097 | -111.477913 | 478109.733333 | 80116.400000 | 52.0 |
| **Arkansas** | 34.911163 | -92.437589 | 34.924027 | -92.428029 | 40184.333333 | 6655.253333 | 54.3 |
| **California** | 37.851530 | -120.724312 | 37.821320 | -120.857914 | 682018.017241 | 94919.965517 | 663.2 |

5 rows × 38 columns

Let's split our data into regions so we can examine the data more closely before choosing features to train our model.

In [62]:

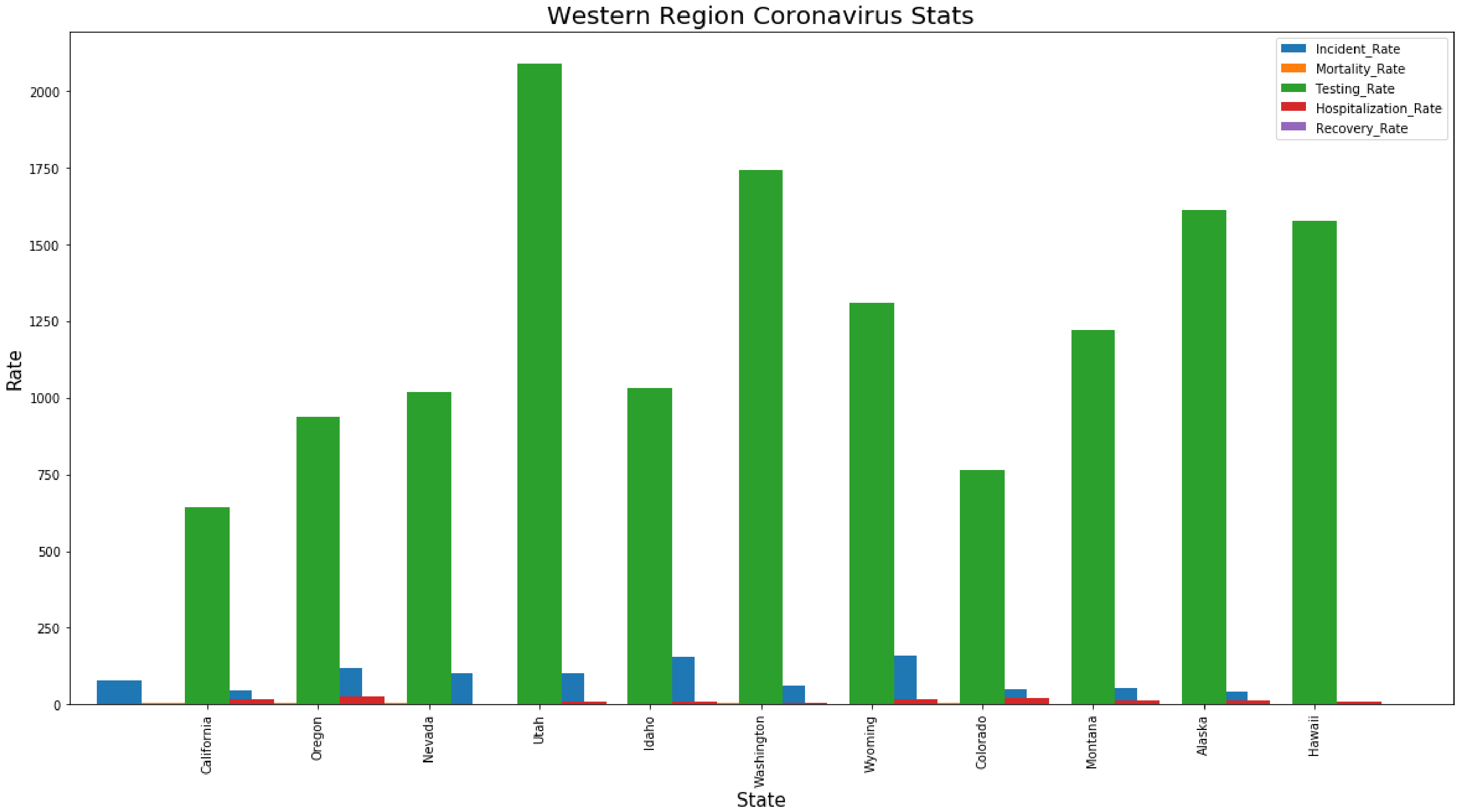
```
west_region = ['California', 'Oregon', 'Nevada', 'Utah', 'Idaho', 'Washington', 'Wyoming', 'Colorado', 'Montana', 'Alaska', 'Hawaii']
southwest_region = ['Arizona', 'New Mexico', 'Oklahoma', 'Texas']
midwest_region = ['North Dakota', 'South Dakota', 'Nebraska', 'Kansas', 'Minnesota', 'Iowa', 'Missouri', 'Wisconsin', 'Illinois', 'Michigan', 'Indiana', 'Ohio']
southeast_region = ['Arkansas', 'Louisiana', 'Mississippi', 'Tennessee', 'Alabama', 'West Virginia', 'Virginia', 'North Carolina', 'South Carolina', 'Georgia', 'Florida']
northeast_region = ['Pennsylvania', 'New York', 'Vermont', 'New Hampshire', 'Massachusetts', 'New Jersey', 'Maryland', 'Delaware', 'Connecticut', 'Rhole Island']
```

In [63]:

```
west_df = join_countries_and_states.loc[west_region, :].reset_index()
southwest_df = join_countries_and_states.loc[southwest_region, :].reset_index()
midwest_df = join_countries_and_states.loc[midwest_region, :].reset_index()
northeast_df = join_countries_and_states.loc[northeast_region, :].reset_index()
southeast_df = join_countries_and_states.loc[southeast_region, :].reset_index()
```

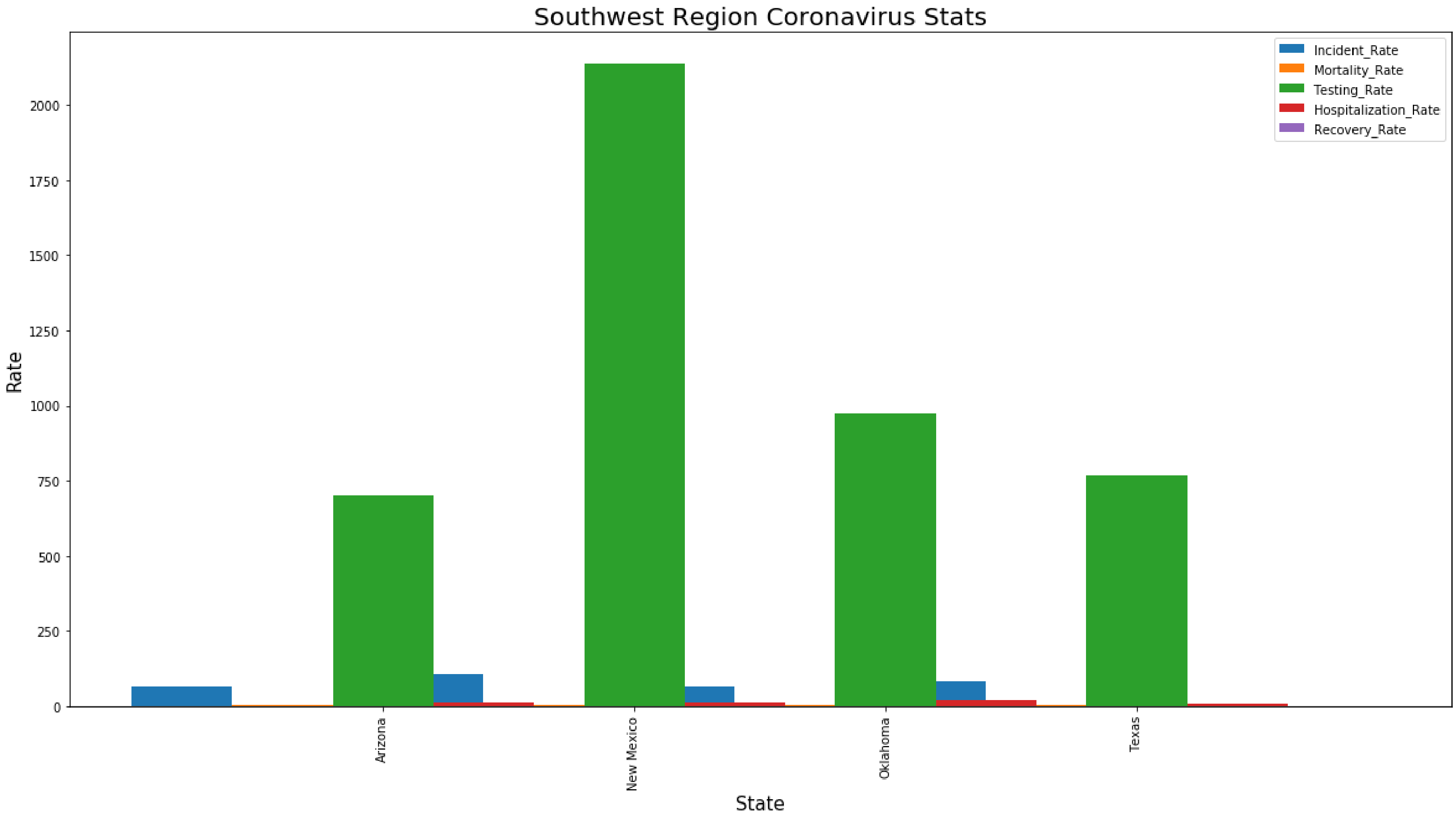
```
/srv/conda/envs/data100/lib/python3.7/site-packages/pandas/core/indexing.py:1418: FutureWarning:


Passing list-likes to .loc or [] with any missing label will raise
KeyError in the future, you can use .reindex() as an alternative.

See the documentation here:
https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#deprecate-loc-reindex-listlike
```

```
In [64]: west_df.loc[:, ['State', 'Incident_Rate', 'Mortality_Rate', 'Testing_Rate', 'Hospitalization_Rate', 'Recovery_
         Rate']].plot.bar(x='State', figsize=(20,10) ,width=2)
         plt.title('Western Region Coronavirus Stats').set_size(20)
         plt.xlabel('State').set_size(15)
         plt.ylabel('Rate').set_size(15)
```



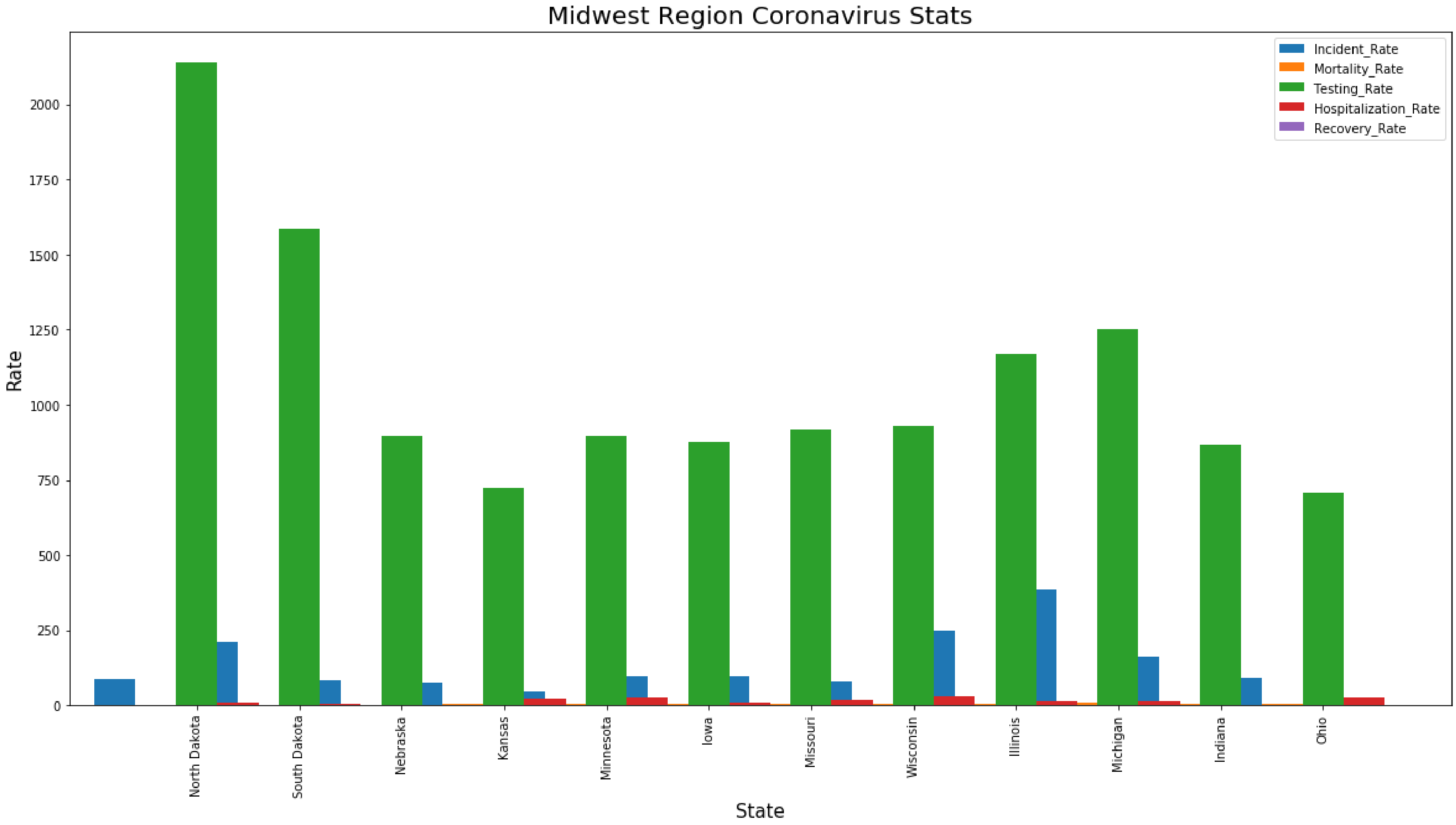
In [65]:

```
southwest_df.loc[:, ['State', 'Incident_Rate', 'Mortality_Rate', 'Testing_Rate', 'Hospitalization_Rate', 'Recovery_Rate']].plot.bar(x='State', figsize=(20,10) ,width=2)
plt.title('Southwest Region Coronavirus Stats').set_size(20)
plt.xlabel('State').set_size(15)
plt.ylabel('Rate').set_size(15)
```

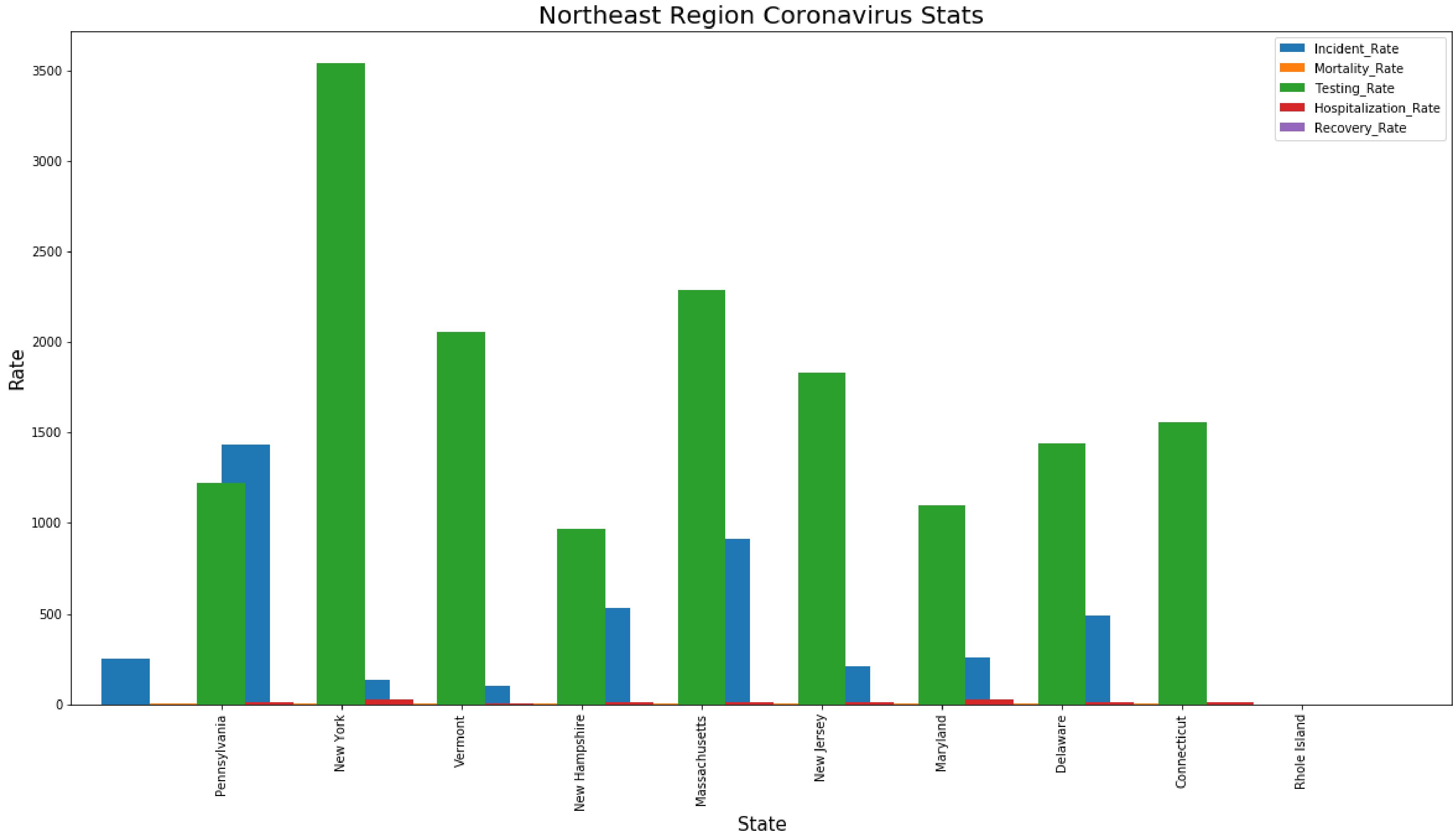


In [66]:

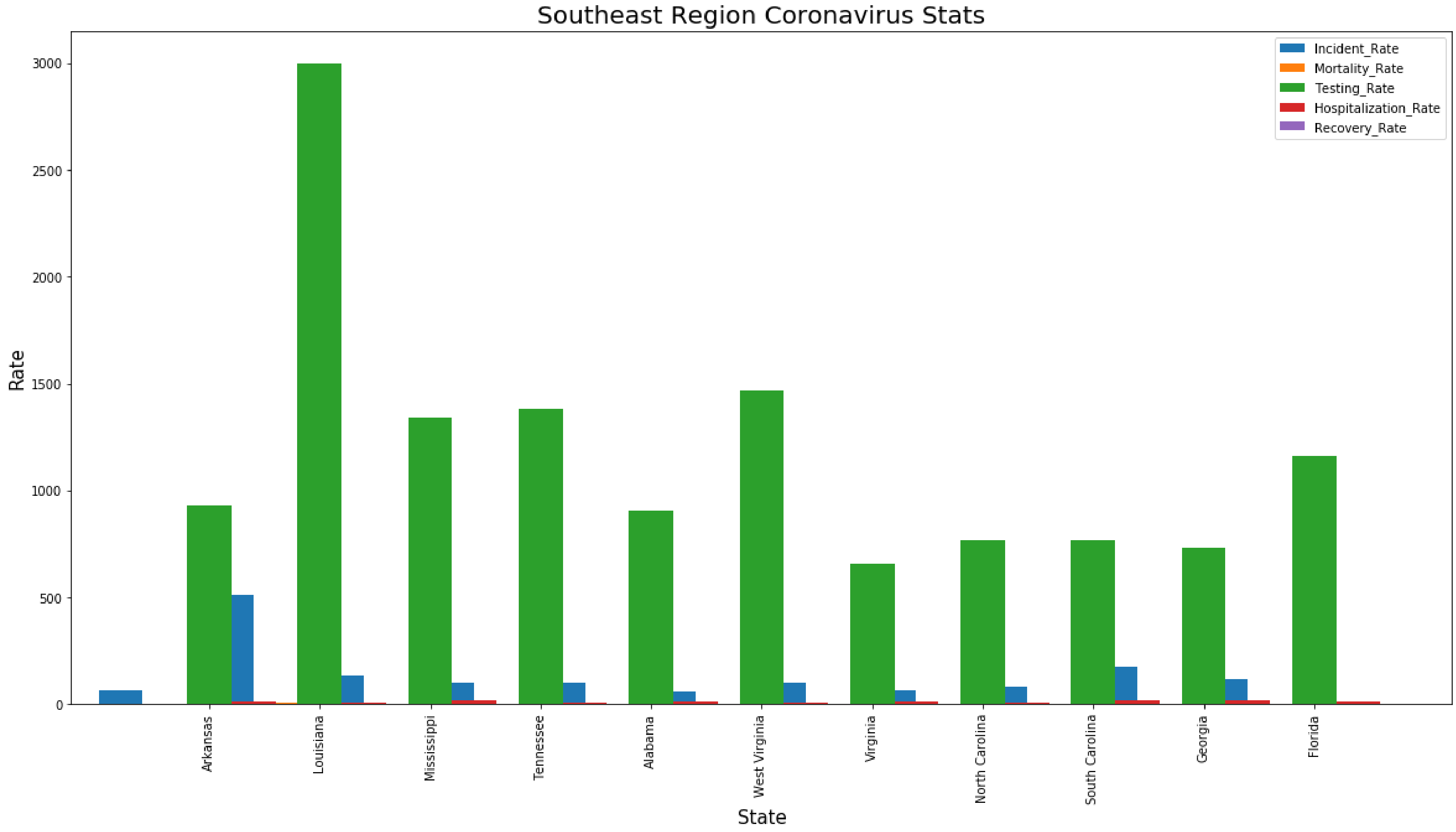
```
midwest_df.loc[:, ['State', 'Incident_Rate', 'Mortality_Rate', 'Testing_Rate', 'Hospitalization_Rate', 'Recovery_Rate']].plot.bar(x='State', figsize=(20,10) ,width=2)
plt.title('Midwest Region Coronavirus Stats').set_size(20)
plt.xlabel('State').set_size(15)
plt.ylabel('Rate').set_size(15)
```



In [67]:
```python
northeast_df.loc[:, ['State', 'Incident_Rate', 'Mortality_Rate', 'Testing_Rate', 'Hospitalization_Rate', 'Recovery_Rate']].plot.bar(x='State', figsize=(20,10) ,width=2)
plt.title('Northeast Region Coronavirus Stats').set_size(20)
plt.xlabel('State').set_size(15)
plt.ylabel('Rate').set_size(15)
```



In [68]:
```
southeast_df.loc[:, ['State', 'Incident_Rate', 'Mortality_Rate', 'Testing_Rate', 'Hospitalization_Rate', 'Recovery_Rate']].plot.bar(x='State', figsize=(20,10) ,width=2)
plt.title('Southeast Region Coronavirus Stats').set_size(20)
plt.xlabel('State').set_size(15)
plt.ylabel('Rate').set_size(15)
```



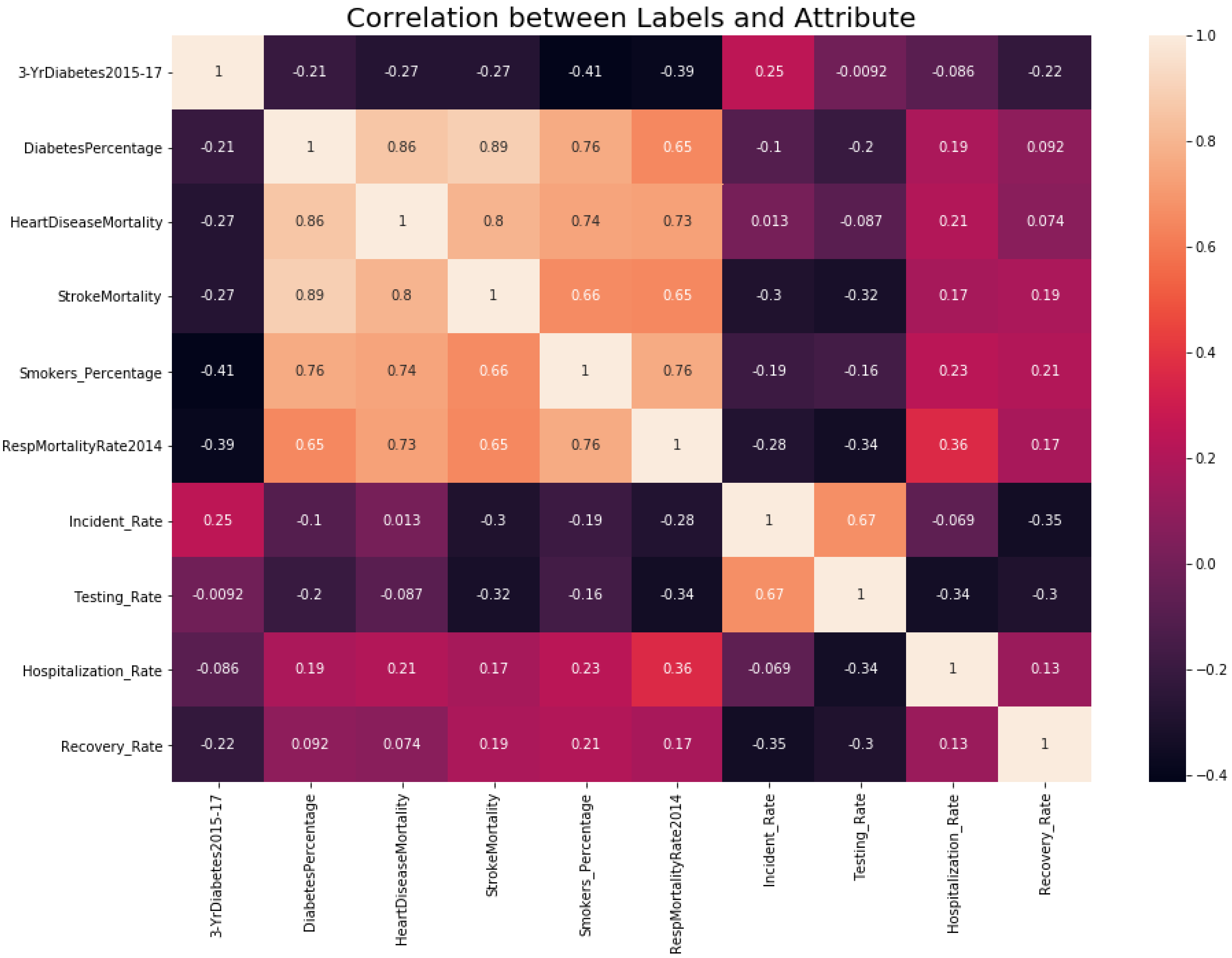
## EXPLORE RELATIONSHIPS

We must divide the data into **attributes** and **labels.**

Labels: `3-YrDiabetes2015-17` , `DiabetesPercentage` , `HeartDiseaseMortality` , `StrokeMortality` , `Smokers_Percentage` , `RespMortalityRate2014` , `Incident_Rate` , `Testing_Rate` , `Hospitalization_Rate`

Attribute: `Recovery_Rate`

In [69]:
```
relationships = join_countries_and_states.loc[:, ['3-YrDiabetes2015-17',
                                                   'DiabetesPercentage',
                                                   'HeartDiseaseMortality',
                                                   'StrokeMortality',
                                                   'Smokers_Percentage',
                                                   'RespMortalityRate2014',
                                                   'Incident_Rate',
                                                   'Testing_Rate',
                                                   'Hospitalization_Rate',
                                                   'Recovery_Rate']]
relationships['Hospitalization_Rate']=relationships['Hospitalization_Rate'].fillna(np.mean(relationships['Hospitalization_Rate']))
corr = relationships.corr()
plt.figure(figsize=(15,10))
sns.heatmap(corr, annot=True)
plt.title('Correlation between Labels and Attribute').set_size(20)
```

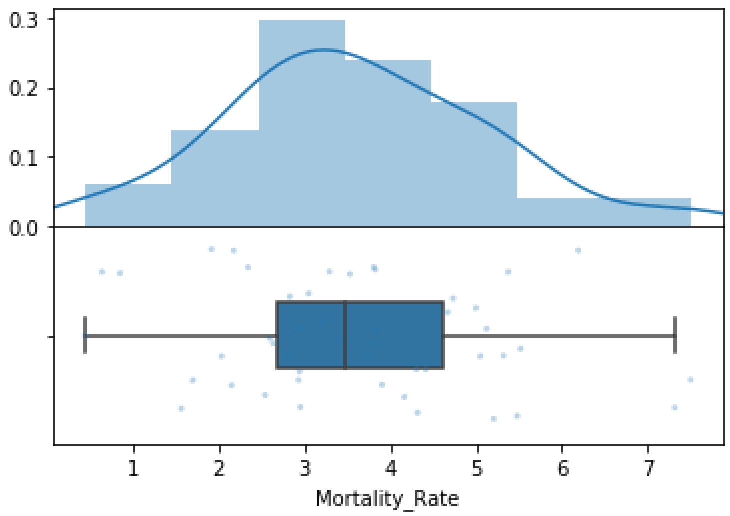


Correlation between Labels and Attribute

Let's further explore the relationships between the labels and attribute using a **raincloud plot** (combination of a KDE, histogram, strip plot, and box plot.) target variable: Recovery_Rate.

In [70]:

```
fig, axs = plt.subplots(nrows=2)

sns.distplot(
    join_countries_and_states['Mortality_Rate'],
    ax=axs[0]
)
sns.stripplot(
    join_countries_and_states['Mortality_Rate'],
    jitter=0.4,
    size=3,
    ax=axs[1],
    alpha=0.3
)
sns.boxplot(
    join_countries_and_states['Mortality_Rate'],
    width=0.3,
    ax=axs[1],
    showfliers=False,
)
spacer = np.max(join_countries_and_states['Mortality_Rate']) * 0.05
xmin = np.min(join_countries_and_states['Mortality_Rate']) - spacer
xmax = np.max(join_countries_and_states['Mortality_Rate']) + spacer
axs[0].set_xlim((xmin, xmax))
axs[1].set_xlim((xmin, xmax))

plt.subplots_adjust(hspace=0)
```



In [71]:

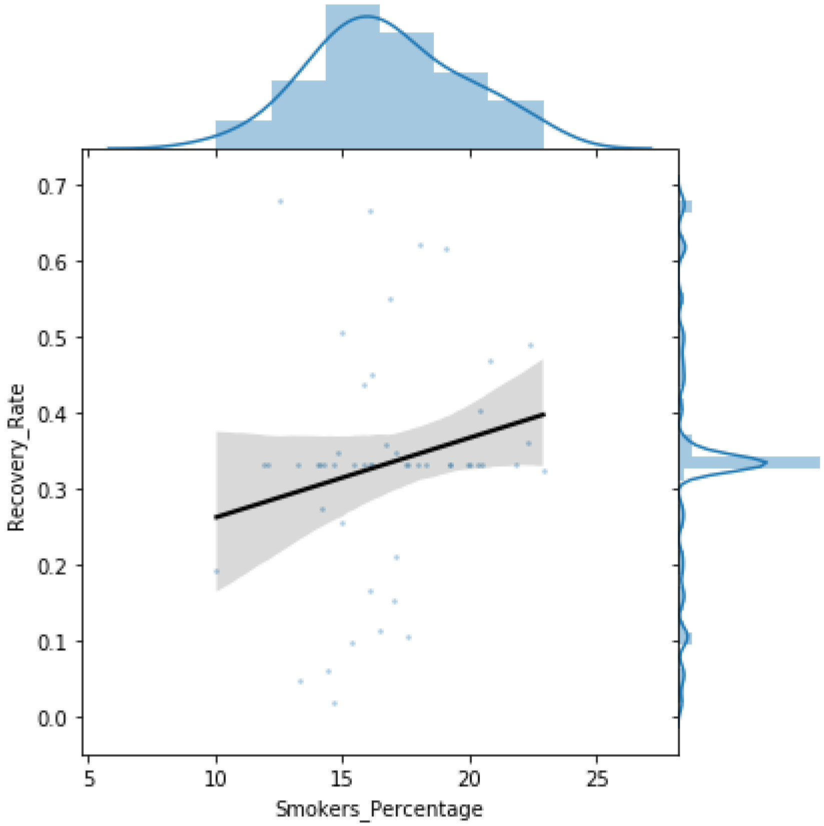
```
join_countries_and_states['Mortality_Rate'].describe()
```

Out[71]:

```
count    50.000000
mean      3.601043
std       1.529845
min       0.453956
25%       2.680235
50%       3.473322
75%       4.603837
max       7.495697
Name: Mortality_Rate, dtype: float64
```

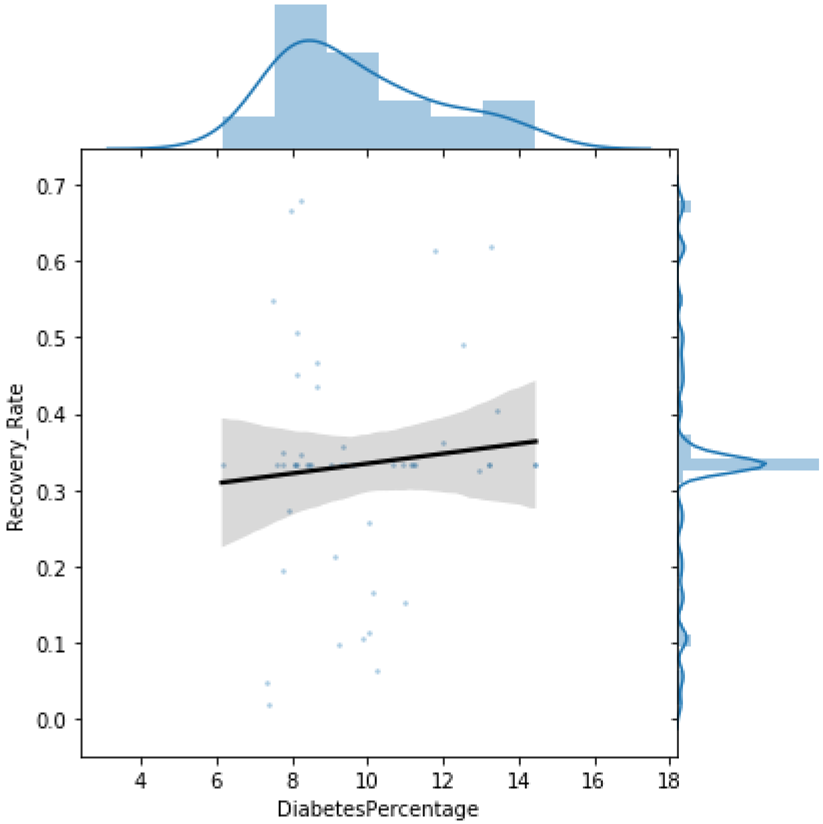
In [72]:

```
sns.jointplot(
    x='Smokers_Percentage',
    y='Recovery_Rate',
    data=join_countries_and_states,
    stat_func=None,
    kind="reg",
    ratio=4,
    space=0,
    scatter_kws={
        's': 3,
        'alpha': 0.25
    },
    line_kws={
        'color': 'black'
    }
);
```

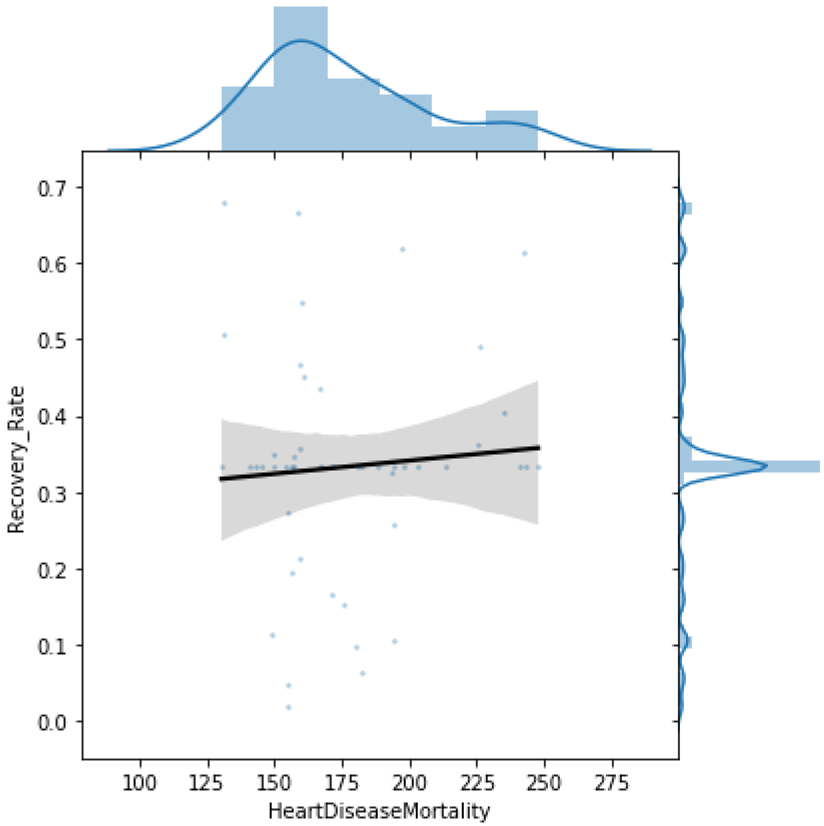

0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
Recovery_Rate
5
10
15
20
25
Smokers_Percentage

In [73]:

```
sns.jointplot(
    x='DiabetesPercentage',
    y='Recovery_Rate',
    data=join_countries_and_states,
    stat_func=None,
    kind="reg",
    ratio=4,
    space=0,
    scatter_kws={
        's': 3,
        'alpha': 0.25
    },
    line_kws={
        'color': 'black'
    }
);
```

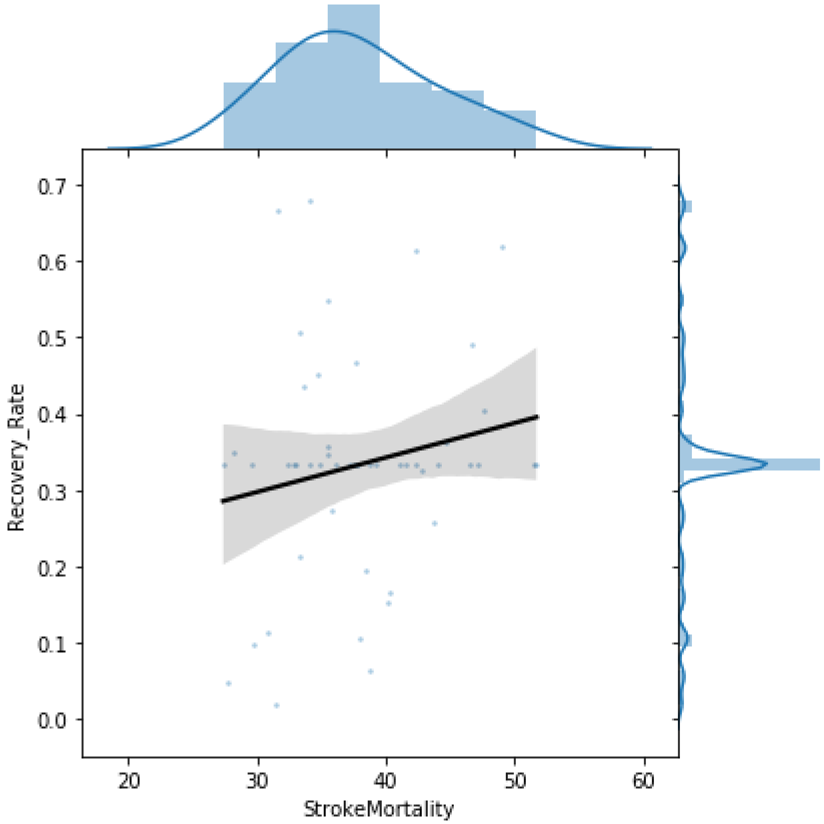

0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
Recovery_Rate
4
6
8
10
12
14
16
18
DiabetesPercentage

```
In [74]: sns.jointplot(
             x='HeartDiseaseMortality',
             y='Recovery_Rate',
             data=join_countries_and_states,
             stat_func=None,
             kind="reg",
             ratio=4,
             space=0,
             scatter_kws={
                 's': 3,
                 'alpha': 0.25
             },
             line_kws={
                 'color': 'black'
             }
         );
```


0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
Recovery_Rate
100
125
150
175
200
225
250
275
HeartDiseaseMortality

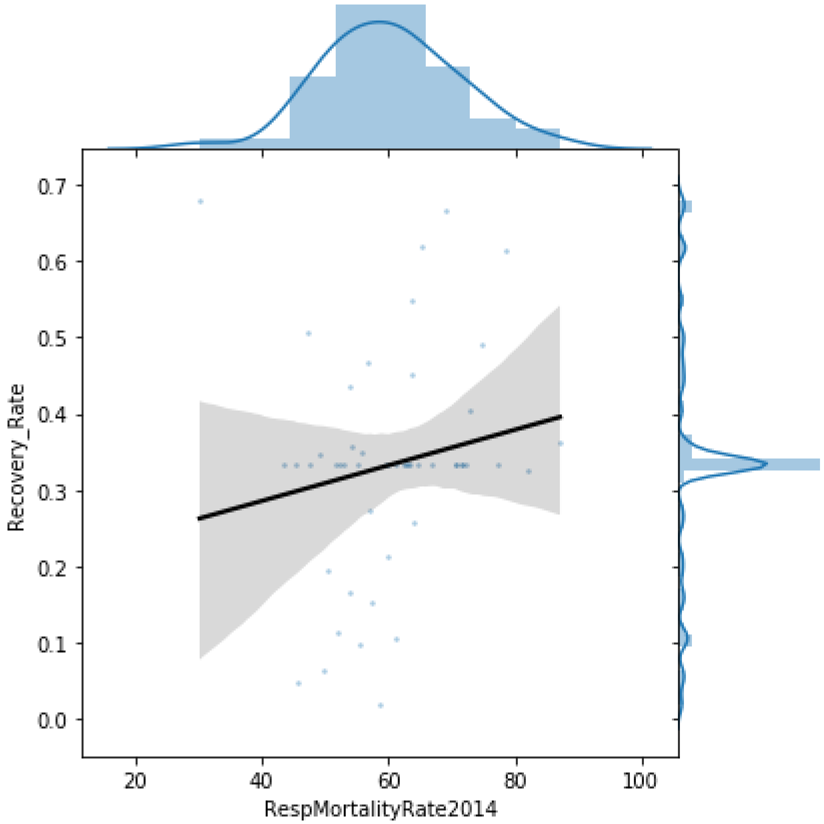
In [75]:

```
sns.jointplot(
    x='StrokeMortality',
    y='Recovery_Rate',
    data=join_countries_and_states,
    stat_func=None,
    kind="reg",
    ratio=4,
    space=0,
    scatter_kws={
        's': 3,
        'alpha': 0.25
    },
    line_kws={
        'color': 'black'
    }
);
```


0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
Recovery_Rate
20
30
40
50
60
StrokeMortality

In [76]:

```
sns.jointplot(
    x='RespMortalityRate2014',
    y='Recovery_Rate',
    data=join_countries_and_states,
    stat_func=None,
    kind="reg",
    ratio=4,
    space=0,
    scatter_kws={
        's': 3,
        'alpha': 0.25
    },
    line_kws={
        'color': 'black'
    }
);
```



## MODEL

```
In [77]: X = join_countries_and_states.loc[:, ['3-YrDiabetes2015-17', 'DiabetesPercentage', 'HeartDiseaseMortality', 'StrokeMortality', 'Smokers_Percentage', 'RespMortalityRate2014', 'Incident_Rate', 'Testing_Rate', 'Hospitalization_Rate']]
X['Hospitalization_Rate']=X['Hospitalization_Rate'].fillna(np.mean(X['Hospitalization_Rate']))
y = join_countries_and_states['Recovery_Rate']
```

In [78]:

```
X.head()
```

Out[78]:

| | 3-YrDiabetes2015-17 | DiabetesPercentage | HeartDiseaseMortality | StrokeMortality | Smokers_Percentage | RespMortalityRate2014 | Incident_Rate |
|---|---|---|---|---|---|---|---|
| **State** | | | | | | | |
| **Alabama** | 31.437500 | 14.407463 | 243.595522 | 51.450746 | 19.989231 | 77.282985 | 100.492717 |
| **Alaska** | 26.250000 | 8.667857 | 159.250000 | 37.659091 | 20.806449 | 56.793448 | 52.530410 |
| **Arizona** | 146.428571 | 10.060000 | 148.826667 | 30.900000 | 16.483911 | 51.968667 | 64.901548 |
| **Arkansas** | 29.833333 | 13.432000 | 235.172000 | 47.681333 | 20.388849 | 72.727067 | 67.361213 |
| **California** | 207.318182 | 8.505172 | 153.908621 | 37.891379 | 12.091600 | 52.153621 | 77.766063 |

In [79]:

```
train, test = train_test_split(join_countries_and_states, test_size=0.2, random_state=42)
```

In [80]:

```
X_train = train.loc[:, ['3-YrDiabetes2015-17', 'DiabetesPercentage', 'HeartDiseaseMortality', 'StrokeMortality', 'Smokers_Percentage', 'RespMortalityRate2014', 'Incident_Rate', 'Testing_Rate', 'Hospitalization_Rate']]
X_train['Hospitalization_Rate']=X_train['Hospitalization_Rate'].fillna(np.mean(X_train['Hospitalization_Rate']))

Y_train = train['Recovery_Rate']
```

In [81]:

```
def normalize(data):
    '''
    Args:
        data : a dataframe
    Returns:
        the normalized version of input data with NAN values filled with 0's
    '''
    new_df = data.copy()
    for i in range(len(data.columns)):
        std = np.std(data[data.columns[i]])
        mean = np.mean(data[data.columns[i]])
        for j in range(len(data[data.columns[i]])):
            x = data[data.columns[i]][j]
            if (std == 0):
                new_df[new_df.columns[i]] = 0
            else:
                new_df[new_df.columns[i]][j] = (x - mean) / std
            j+=1
        i+=1
    return new_df
```

In [82]:

```
X_train_df = pd.DataFrame(X_train)
X_train = normalize(X_train_df)
```

In [83]:

```
X_test = test.loc[:, ['3-YrDiabetes2015-17', 'DiabetesPercentage', 'HeartDiseaseMortality', 'StrokeMortality',
'Smokers_Percentage', 'RespMortalityRate2014', 'Incident_Rate', 'Testing_Rate', 'Hospitalization_Rate']]
X_test['Hospitalization_Rate']=X_test['Hospitalization_Rate'].fillna(np.mean(X_test['Hospitalization_Rate']))

Y_test = test['Recovery_Rate']
```

In [84]:

```
X_test.head()
```

Out[84]:

| State | 3-YrDiabetes2015-17 | DiabetesPercentage | HeartDiseaseMortality | StrokeMortality | Smokers_Percentage | RespMortalityRate2014 | Incident_Rate |
|---|---|---|---|---|---|---|---|
| **Indiana** | 33.537037 | 11.191304 | 188.691304 | 41.625000 | 19.932077 | 70.725109 | 162.607117 |
| **South Carolina** | 39.235294 | 13.267391 | 197.263043 | 49.047826 | 18.045662 | 65.362391 | 84.286132 |
| **New Mexico** | 33.222222 | 9.157576 | 159.251515 | 33.366667 | 17.066621 | 59.979697 | 107.823953 |
| **Virginia** | 28.473684 | 10.983459 | 175.609023 | 40.103008 | 16.991193 | 57.262180 | 101.843503 |
| **Louisiana** | 34.406250 | 13.232812 | 241.418750 | 47.168750 | 21.792924 | 62.449688 | 512.913545 |

In [85]:

```
X_test_df = pd.DataFrame(X_test)
X_test = normalize(X_test_df)
```

In [86]:

```
model = LinearRegression()
model.fit(X_train, Y_train)
Y_pred = model.predict(X_test)
```

```
In [87]: Y_test
```

```
Out[87]: State
         Indiana             0.333147
         South Carolina      0.619821
         New Mexico          0.212458
         Virginia            0.152490
         Louisiana           0.333147
         Wisconsin           0.333147
         Nebraska            0.333147
         Montana             0.549296
         North Carolina      0.333147
         Maryland            0.062551
         Name: Recovery_Rate, dtype: float64
```

```
In [88]: Y_pred.tolist()
```

```
Out[88]: [0.2708828245566953,
          0.31578603195037586,
          0.2827747159922025,
          0.4172656091739701,
          0.20151366215404656,
          0.5389913079338842,
          0.331777186812271,
          0.3497514815884232,
          0.2685228490788606,
          0.37148445162306354]
```

```
In [89]: metrics.mean_absolute_error(Y_test, Y_pred)
```

```
Out[89]: 0.16133412405652992
```

```
In [90]: metrics.mean_squared_error(Y_test, Y_pred)
```

```
Out[90]: 0.037050015723393295
```

```
In [91]: np.sqrt(metrics.mean_squared_error(Y_test, Y_pred))
```

```
Out[91]: 0.19248380639262436
```

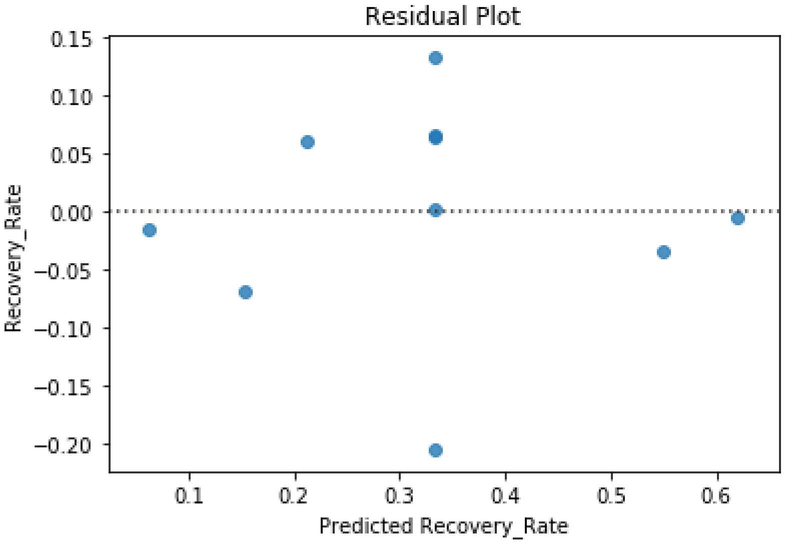
In [93]:

```
sns.residplot(Y_test, Y_test-Y_pred, lowess=True)
plt.title('Residual Plot')
plt.xlabel('Actual Recovery_Rate')
plt.xlabel('Predicted Recovery_Rate')
```

```
/srv/conda/envs/data100/lib/python3.7/site-packages/statsmodels/nonparametric/smoothers_lowess.py:165: Runti
meWarning:

invalid value encountered in greater_equal
```

Out[93]:

```
Text(0.5, 0, 'Predicted Recovery_Rate')
```



In [94]:

```
from sklearn.model_selection import KFold

kf = KFold(n_splits=4)
kf.get_n_splits(X_train)
KFold(n_splits = 4, random_state = 42, shuffle = False)

from sklearn.model_selection import cross_val_score, cross_val_predict
from sklearn import metrics
scores = cross_val_score(model, X_train, Y_train, cv=6, scoring='neg_mean_squared_error')
scores
```

```
/srv/conda/envs/data100/lib/python3.7/site-packages/sklearn/model_selection/_split.py:296: FutureWarning:

Setting a random_state has no effect since shuffle is False. This will raise an error in 0.24. You should le
ave random_state to its default (None), or set shuffle=True.
```

Out[94]:

```
array([-0.01103514, -0.02494853, -0.05712099, -0.04246981, -0.06255325,
       -0.01658819])
```

In [95]:

```
X['Hospitalization_Rate'] = X['Hospitalization_Rate'].fillna(np.mean(X['Hospitalization_Rate']))
join_countries_and_states['Predicted Recovery Rates'] = model.predict(normalize(X))
join_countries_and_states.loc[:, ['Recovery_Rate', 'Predicted Recovery Rates']]
```

Out[95]:

| | Recovery_Rate | Predicted Recovery Rates |
|---|---|---|
| **State** | | |
| **Alabama** | 0.333147 | 0.317520 |
| **Alaska** | 0.468153 | 0.422385 |
| **Arizona** | 0.114098 | 0.316995 |
| **Arkansas** | 0.403096 | 0.382509 |
| **California** | 0.333147 | 0.255202 |
| **Colorado** | 0.333147 | 0.343062 |
| **Connecticut** | 0.333147 | 0.251295 |
| **Delaware** | 0.166667 | 0.292256 |
| **Florida** | 0.333147 | 0.302038 |
| **Georgia** | 0.333147 | 0.330098 |
| **Hawaii** | 0.679443 | 0.419838 |
| **Idaho** | 0.273716 | 0.341614 |
| **Illinois** | 0.333147 | 0.299163 |
| **Indiana** | 0.333147 | 0.340839 |
| **Iowa** | 0.435734 | 0.395865 |
| **Kansas** | 0.333147 | 0.390900 |
| **Kentucky** | 0.361655 | 0.426445 |
| **Louisiana** | 0.333147 | 0.246747 |
| **Maine** | 0.451004 | 0.351938 |
| **Maryland** | 0.062551 | 0.412817 |
| **Massachusetts** | 0.333147 | 0.202540 |
| **Michigan** | 0.105128 | 0.316050 |
| **Minnesota** | 0.506111 | 0.493486 |

| | Recovery_Rate | Predicted Recovery Rates |
|---:|---:|---:|
| **State** | | |
| **Mississippi** | 0.333147 | 0.376724 |
| **Missouri** | 0.333147 | 0.388777 |
| **Montana** | 0.549296 | 0.368553 |
| **Nebraska** | 0.333147 | 0.362522 |
| **Nevada** | 0.333147 | 0.319818 |
| **New Hampshire** | 0.348733 | 0.385368 |
| **New Jersey** | 0.333147 | 0.228383 |
| **New Mexico** | 0.212458 | 0.305412 |
| **New York** | 0.098824 | 0.068168 |
| **North Carolina** | 0.333147 | 0.315254 |
| **North Dakota** | 0.346591 | 0.385804 |
| **Ohio** | 0.333147 | 0.430326 |
| **Oklahoma** | 0.614604 | 0.379442 |
| **Oregon** | 0.333147 | 0.387219 |
| **Pennsylvania** | 0.333147 | 0.364085 |
| **Rhode Island** | 0.048319 | 0.225341 |
| **South Carolina** | 0.619821 | 0.367216 |
| **South Dakota** | 0.357977 | 0.317775 |
| **Tennessee** | 0.490818 | 0.334176 |
| **Texas** | 0.256950 | 0.304318 |
| **Utah** | 0.193692 | 0.257791 |
| **Vermont** | 0.018680 | 0.283603 |
| **Virginia** | 0.152490 | 0.419200 |
| **Washington** | 0.333147 | 0.229023 |

| | Recovery_Rate | Predicted Recovery Rates |
|---|---|---|
| **State** | | |
| **West Virginia** | 0.324841 | 0.251634 |
| **Wisconsin** | 0.333147 | 0.512432 |
| **Wyoming** | 0.666667 | 0.323784 |

## Line Plot Overtime From Dates 4-19-2020 to 5-10-2020

In [96]:

```python
april_19 = pd.read_csv("04-19-2020.csv")
april_20 = pd.read_csv("04-20-2020.csv")
april_21 = pd.read_csv("04-21-2020.csv")
april_22 = pd.read_csv("04-22-2020.csv")
april_23 = pd.read_csv("04-23-2020.csv")
april_24 = pd.read_csv("04-24-2020.csv")
april_25 = pd.read_csv("04-25-2020.csv")
april_26 = pd.read_csv("04-26-2020.csv")
april_27 = pd.read_csv("04-27-2020.csv")
april_28 = pd.read_csv("04-28-2020.csv")
april_29 = pd.read_csv("04-29-2020.csv")
april_30 = pd.read_csv("04-30-2020.csv")
```

In [97]:

```python
may_1 = pd.read_csv("05-01-2020.csv")
may_2 = pd.read_csv("05-02-2020.csv")
may_3 = pd.read_csv("05-03-2020.csv")
may_4 = pd.read_csv("05-04-2020.csv")
may_5 = pd.read_csv("05-05-2020.csv")
may_6 = pd.read_csv("05-06-2020.csv")
may_7 = pd.read_csv("05-07-2020.csv")
may_8 = pd.read_csv("05-08-2020.csv")
may_9 = pd.read_csv("05-09-2020.csv")
may_10 = pd.read_csv("05-10-2020.csv")
```

In [98]:

```
dates = [april_19, april_20, april_21, april_22, april_23, april_24, april_25,
         april_26, april_27, april_28, april_29, april_30, may_1, may_2, may_3,
         may_4, may_5, may_6, may_7, may_8, may_9, may_10]

all_dates = pd.concat(dates)
all_dates
```

Out[98]:

| | Province_State | Country_Region | Last_Update | Lat | Long_ | Confirmed | Deaths | Recovered | Active | FIPS | Incident_Rate | People_Tested |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | Alabama | US | 2020-04-19 23:41:01 | 32.3182 | -86.9023 | 4888 | 157 | NaN | 4731.0 | 1.0 | 104.246265 | 45712.0 |
| **1** | Alaska | US | 2020-04-19 23:41:01 | 61.3707 | -152.4044 | 319 | 9 | 153.0 | 310.0 | 2.0 | 53.366881 | 9895.0 |
| **2** | American Samoa | US | NaN | -14.2710 | -170.1320 | 0 | 0 | NaN | NaN | 60.0 | 0.000000 | 3.0 |
| **3** | Arizona | US | 2020-04-19 23:41:01 | 33.7298 | -111.4312 | 4933 | 184 | 994.0 | 4749.0 | 4.0 | 67.772933 | 52990.0 |
| **4** | Arkansas | US | 2020-04-19 23:41:01 | 34.9697 | -92.3731 | 1781 | 39 | 721.0 | 1742.0 | 5.0 | 68.790322 | 24209.0 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **53** | Virginia | US | 2020-05-11 02:32:34 | 37.7693 | -78.1700 | 24081 | 839 | 3201.0 | 20041.0 | 51.0 | 304.544069 | 143055.0 |
| **54** | Washington | US | 2020-05-11 02:32:34 | 47.4009 | -121.4905 | 16891 | 931 | NaN | 15960.0 | 53.0 | 223.739546 | 242989.0 |
| **55** | West Virginia | US | 2020-05-11 02:32:34 | 38.4912 | -80.9545 | 1360 | 54 | 775.0 | 531.0 | 54.0 | 102.796830 | 62644.0 |
| **56** | Wisconsin | US | 2020-05-11 02:32:34 | 44.2685 | -89.6165 | 10219 | 400 | 5014.0 | 4805.0 | 55.0 | 197.484970 | 115382.0 |
| **57** | Wyoming | US | 2020-05-11 02:32:34 | 42.7560 | -107.3025 | 662 | 7 | 443.0 | 212.0 | 56.0 | 133.134905 | 12064.0 |

1368 rows × 18 columns

In [99]:

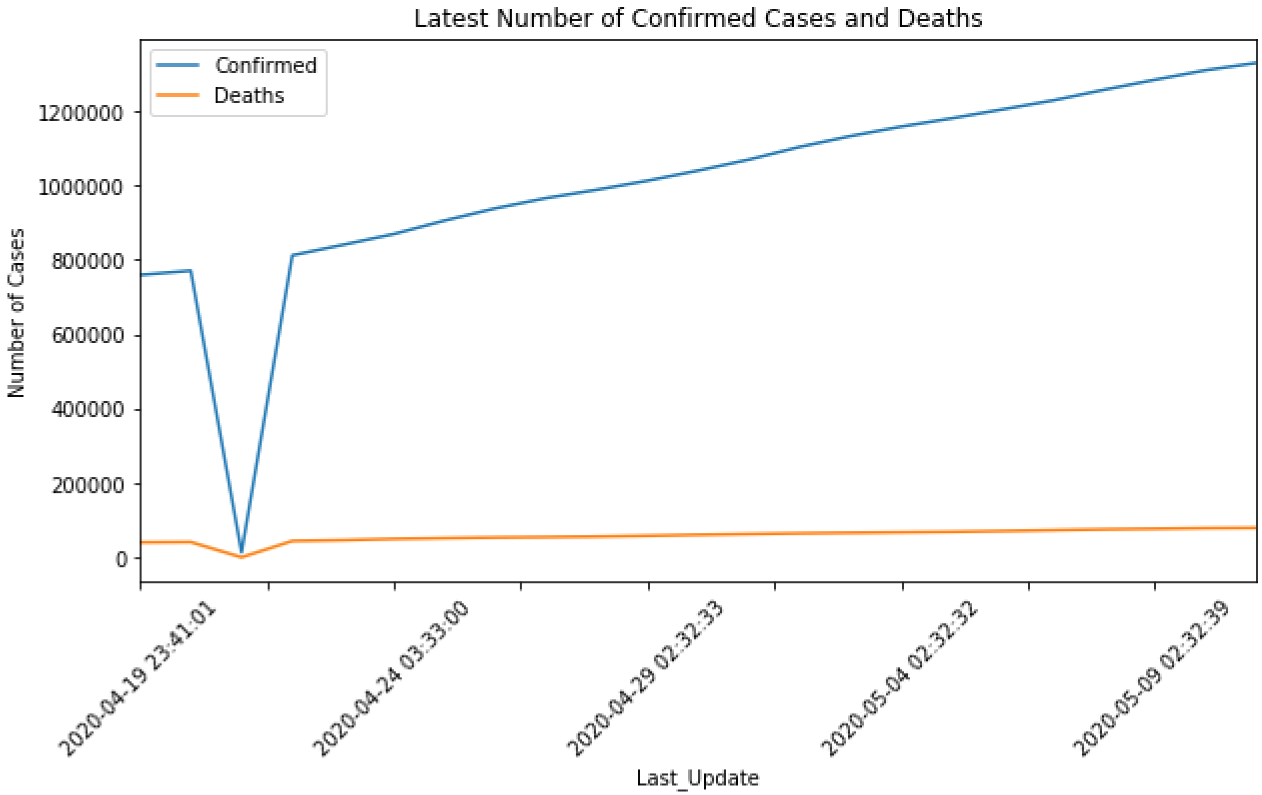
```
grouped_by_date = all_dates.groupby('Last_Update').sum()
grouped_by_date = grouped_by_date.loc[:, ['Confirmed', 'Deaths']]
```

In [100]:

```python
grouped_by_date.plot(figsize = (10, 5))
plt.xticks(rotation=45)
plt.ylabel('Number of Cases')
plt.title('Latest Number of Confirmed Cases and Deaths')
```

Out[100]:

```
Text(0.5, 1.0, 'Latest Number of Confirmed Cases and Deaths')
```


Latest Number of Confirmed Cases and Deaths
Confirmed
Deaths
Number of Cases
2020-04-19 23:41:01
2020-04-24 03:33:00
2020-04-29 02:32:33
2020-05-04 02:32:32
2020-05-09 02:32:39
Last_Update