databricks Project_Report_V1

Health Care Analytics

Summary:

- Health care industry is huge and there is lot of data coming out of health care.
- Overall spending is $3.8 trillion per year in US health care.
- The estimated waste is $765 billion dollors.
- Not only is cost an issue, but health care is poor.
- In this project we are focusing on analysis of US health care and proving solutions that provide better health care at lower cost.

Data Set:

data.medicare.gov, healthcare.gov

Tools

PySpark, Python (Visualizations)

By

Ekta Shah, Parvathi Pai

Analysis 1: Average cost of insurance in 50 states

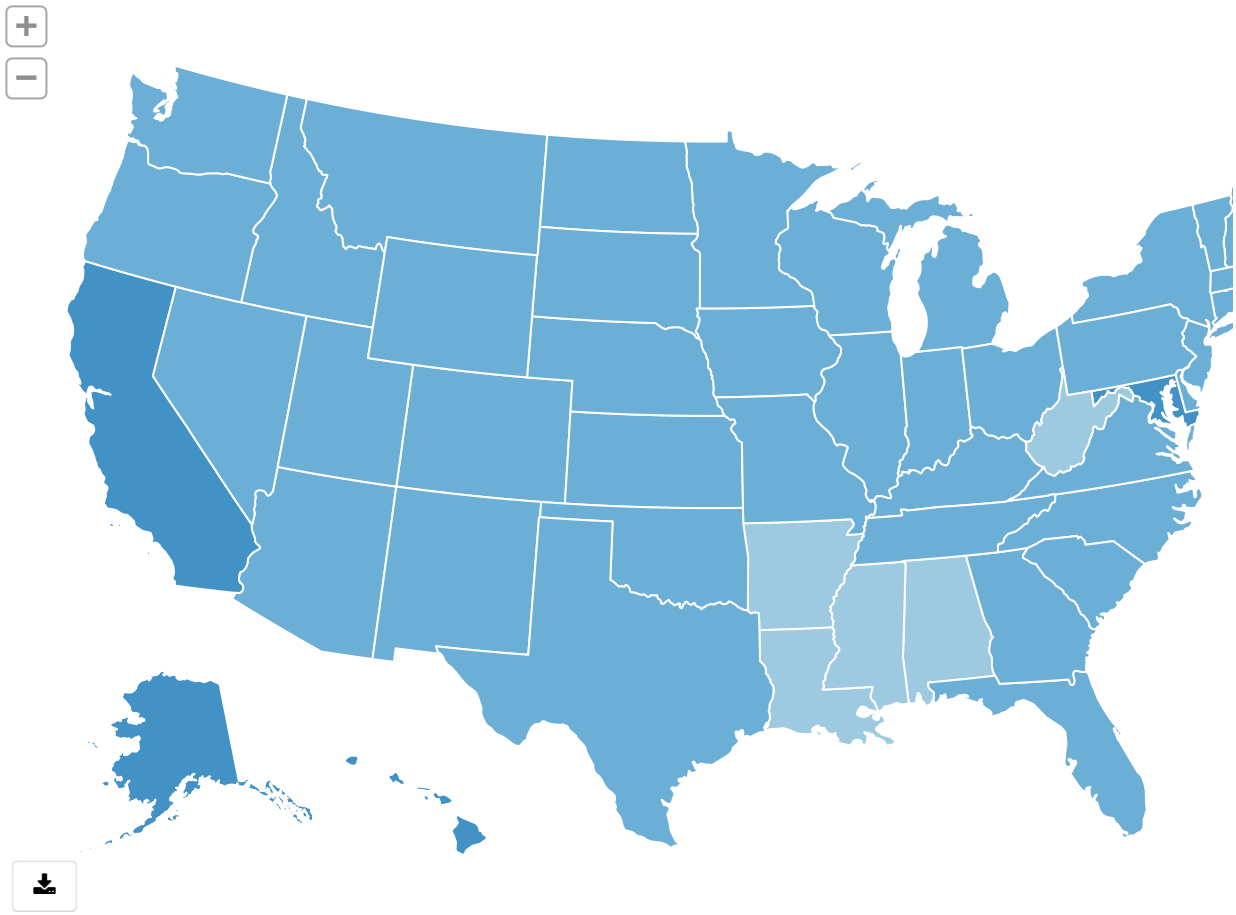
- For this analysis we used data from inpatient prospective payment system.
- The data includes hospital specific charges for 3000 US hospitals which covers 50 states.
- Average covered charge: The average charge billed to the medicare by the provider.
- Average total payments: The total payments made to the provider (including payments from medicare as well as the co-payments and deductibles paid by the beneficiary)

- Average Medicare Payments: The average payment just from medicare.
- Variation in insurance cost based on age.

```
%scala
val ippsdf =
spark.read.option("inferSchema","true").option("header","true").csv("dbfs:/autu
mn_2019/ektashah/ProjectDataSet/IPPS_2017.csv")
import org.apache.spark.sql.functions._
val a_ippdf_1 = ippsdf.groupBy(($"Provider State") as "ProviderState"
).agg(avg($"Average Medicare Payments") as
"AverageMedicarePayment",avg($"Average Total Payments") as
"AverageTotalPayment")
val a_ippsdf_2 = ippsdf.groupBy(($"DRG Definition") as "DRG Definition",
($"Provider Name") as "ProviderName", ($"Provider State") as "ProviderState"
).agg(avg($"Average Medicare Payments") as
"AverageMedicarePayment",avg($"Average Total Payments") as
"AverageTotalPayment")
```

```
ippsdf: org.apache.spark.sql.DataFrame = [DRG Definition: string, Provider Id:
int ... 10 more fields]
import org.apache.spark.sql.functions._
a_ippdf_1: org.apache.spark.sql.DataFrame = [ProviderState: string, AverageMedi
carePayment: double ... 1 more field]
a_ippsdf_2: org.apache.spark.sql.DataFrame = [DRG Definition: string, ProviderN
ame: string ... 3 more fields]
```

Show code



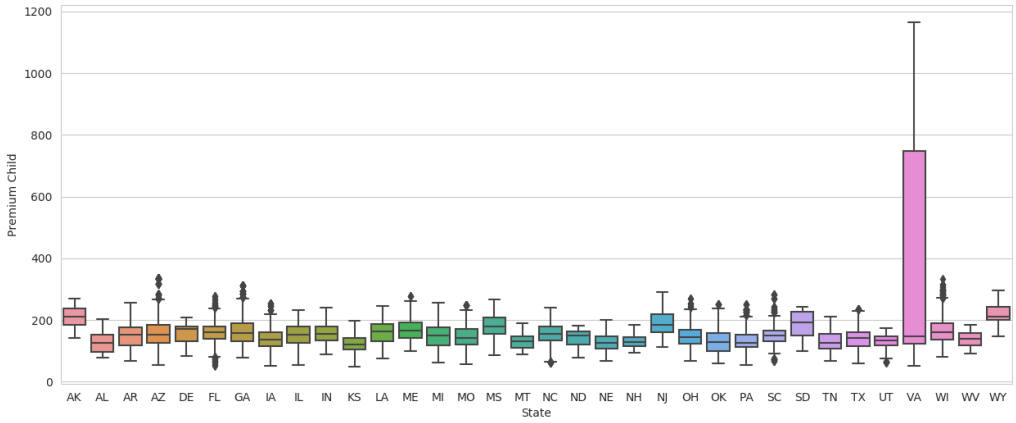
```
import pandas as pd
import seaborn as sns
import numpy as np
import matplotlib.pyplot as plt
df_average_insurance=pd.read_csv('/dbfs/autumn_2019/pava/ProjectData/Insurance.
csv')
df_average_insurance.describe()
```

```
/databricks/python/lib/python3.7/site-packages/IPython/core/interactiveshell.p
y:3049: DtypeWarning: Columns (17) have mixed types. Specify dtype option on im
port or set low_memory=False.
  interactivity=interactivity, compiler=compiler, result=result)
Out[1]:
```

	Premium Scenarios	Premium Child	Premium Adult Individual Age 21	Premium Adult Individual Age 27	Premium Adult Individual Age 30	Premium Adult Individual Age 40	Prei Indiv A
count	0.0	76336.000000	77380.000000	77380.000000	77380.000000	77380.000000	77380.0(
mean	NaN	161.284730	253.229001	266.654782	288.358541	324.371911	453.52
std	NaN	108.366481	169.939068	177.695466	192.536102	216.879044	303.02
min	NaN	49.070000	77.270000	80.980000	87.710000	98.760000	138.01
25%	NaN	120.845000	188.810000	200.490000	216.560000	243.000000	340.27
50%	NaN	146.190000	229.880000	241.950000	261.420000	294.250000	411.21
75%	NaN	174.690000	275.390000	288.910000	312.580000	351.960000	491.86
max	NaN	1164.980000	1834.620000	1922.680000	2082.290000	2344.640000	3276.63

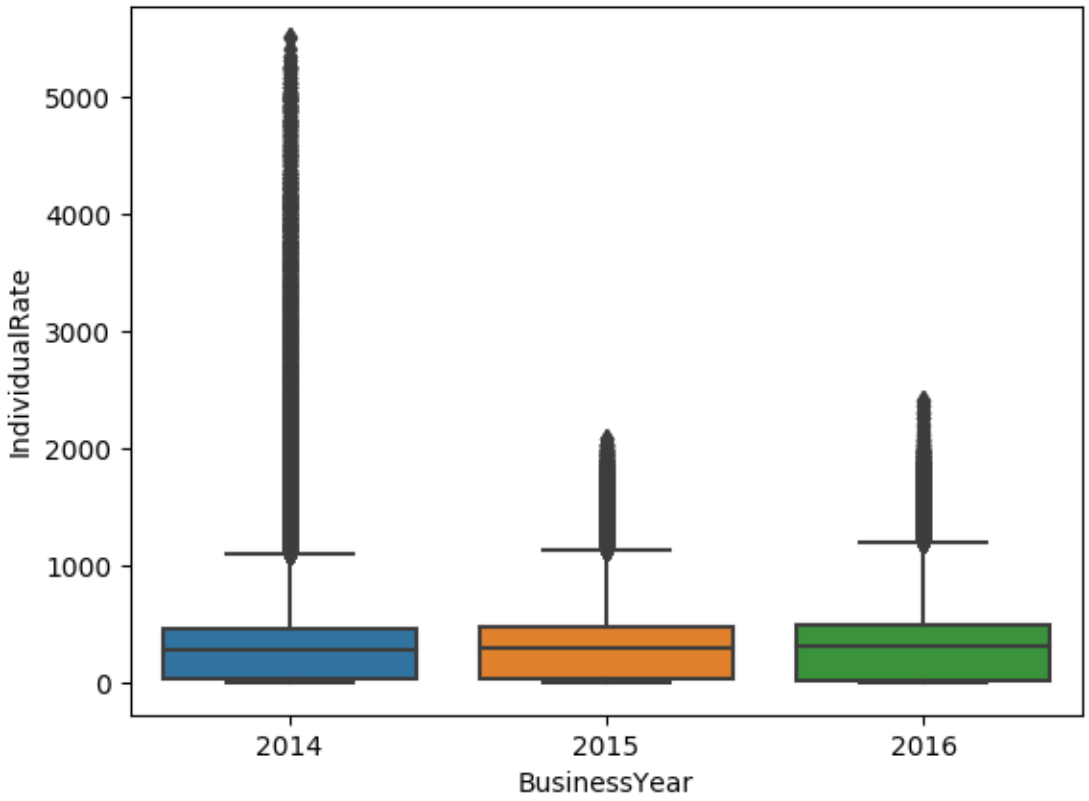
For the same count of subscribers their is variation of insurance with respect to age and children.

```
Statelist = df_average_insurance['State'].unique()
Statelist = np.sort(Statelist)
plt.figure(figsize=(15, 6))
sns.set_style("whitegrid")
sns.boxplot(x="State", y="Premium Child",data=df_average_insurance,
order=Statelist)
display()
```



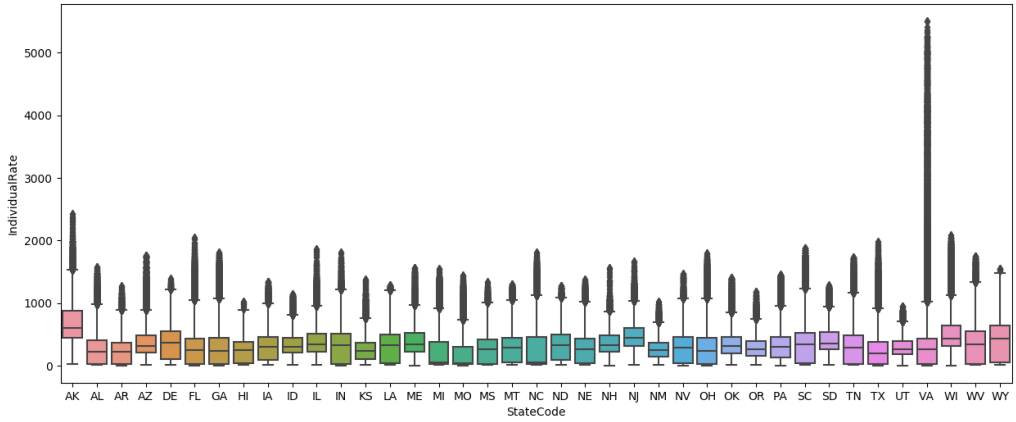
Insurance premium for children is highest fluctuated in VA.

```
import pandas as pd
import seaborn as sns
import numpy as np
import matplotlib.pyplot as plt
df_va=pd.read_csv('/dbfs/autumn_2019/pava/ProjectData/Rate.csv')
na_values = ['NaN', 'N/A', '0', '0.01', '9999', '9999.99', '999999']
df_va = pd.read_csv("/dbfs/autumn_2019/pava/ProjectData/Rate.csv",
na_values=na_values, usecols=['BusinessYear', 'StateCode', 'PlanId',
'RatingAreaId','Tobacco', 'Age',
'IndividualRate','IndividualTobaccoRate','Couple',
'PrimarySubscriberAndOneDependent', 'PrimarySubscriberAndTwoDependents',
'PrimarySubscriberAndThreeOrMoreDependents', 'CoupleAndOneDependent',
'CoupleAndTwoDependents', 'CoupleAndThreeOrMoreDependents'])
df_va = df_va.drop_duplicates().reset_index(drop=True)
sns.boxplot(x="BusinessYear", y="IndividualRate", data=df_va)
display()
```



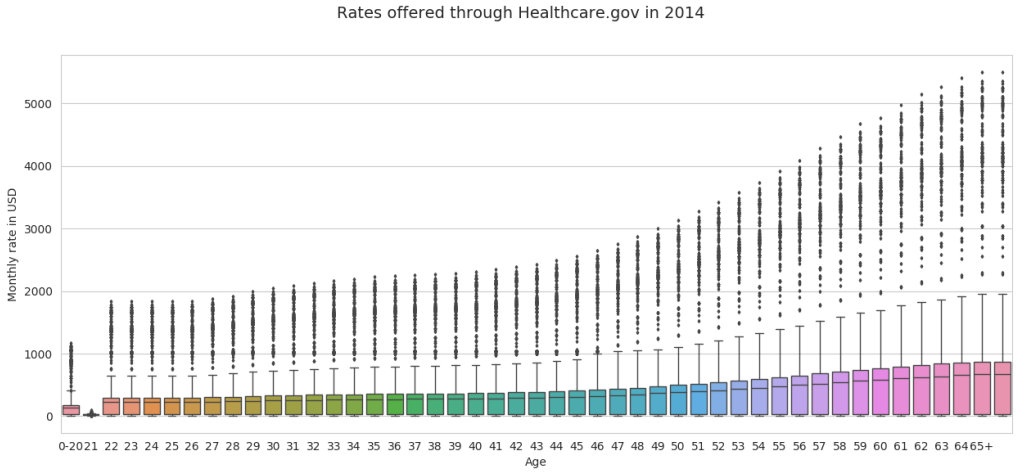
But average insurance rate is same in all three years with some outliers in 2014

```
Statelist = df_va['StateCode'].unique()
Statelist = np.sort(Statelist)
plt.figure(figsize=(15, 6))
sns.boxplot(x="StateCode", y="IndividualRate", data=df_va, order=Statelist)
display()
```



Adult insurance in VA is comparable with other states. Only 75 percentile of poplution pay more for child insurance in VA

```
fig = plt.figure(figsize=(15, 6))
sns.set_style("whitegrid")

fig.suptitle('Rates offered through Healthcare.gov in 2014', fontsize=14)
df2014 = df_va[df_va['BusinessYear'].isin([2014])].copy()
ax = sns.boxplot(x="Age", y="IndividualRate", data=df2014, linewidth=1.0,
fliersize=2.0)
ax.set_ylabel("Monthly rate in USD")
list = df2014['Age'].isin(['Family Option'])
df2014_va_wofamily = df2014[~list]
age_labels = df2014_va_wofamily['Age'].unique()
age_labels = ['65+' if x=='65 and over' else x for x in age_labels]  #replace
label '65 and over' with '65+' for plot labels
len(age_labels), age_labels
xticks = np.arange(46)
ax.xaxis.set_ticks(xticks)
ax.set_xticklabels(age_labels)

plt.savefig('Virginia_rates_by_age.png', bbox_inches='tight', dpi=150)
display()
```



The rate of premium insurance increases with age

Show code

Out[8]:

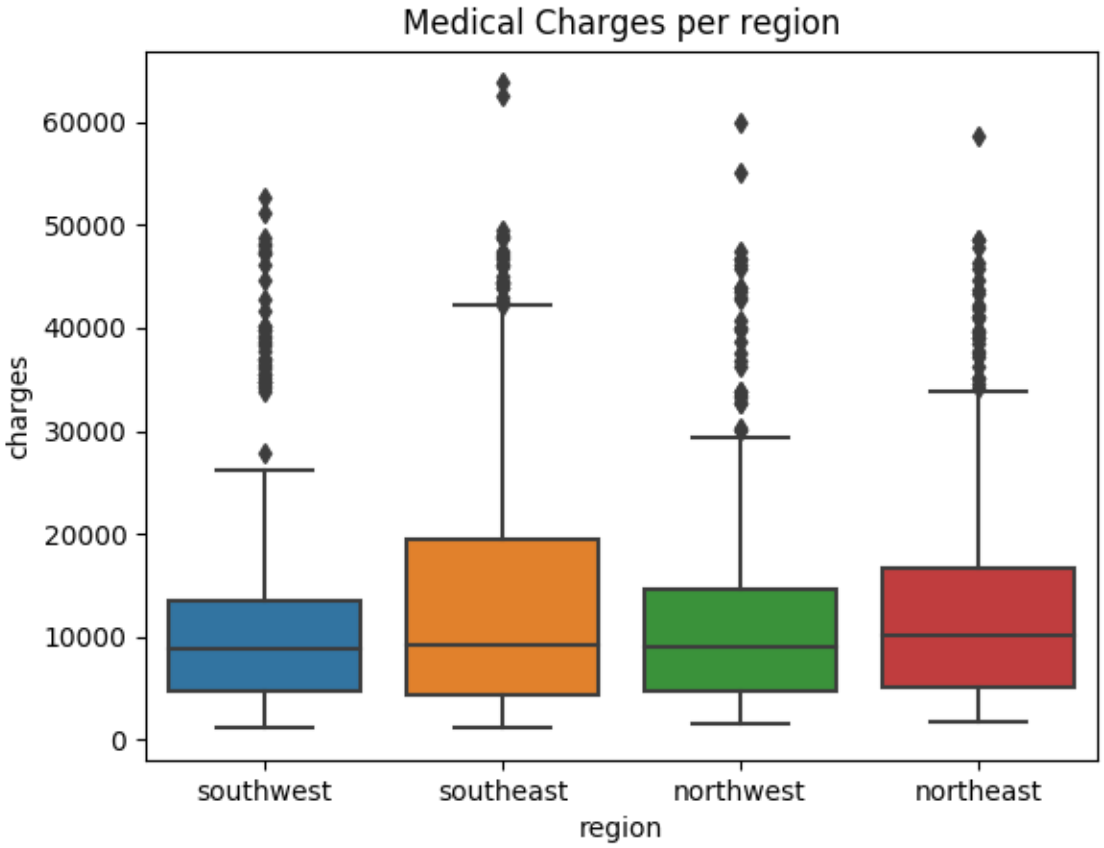
Plan ID - Standard Component	median_Premium Adult Individual Age 21	median_Premium Adult Individual Age 27	median_Premium Adult Individual Age 30	median_Premium Adult Individual Age 40	median_ Adult
38344AK0570001	280.0	294.0	318.0	358.0	
38344AK0570002	333.0	348.0	377.0	425.0	
38344AK0600002	408.0	427.0	463.0	521.0	

The insurance cost increases with age by 5 to 10%.

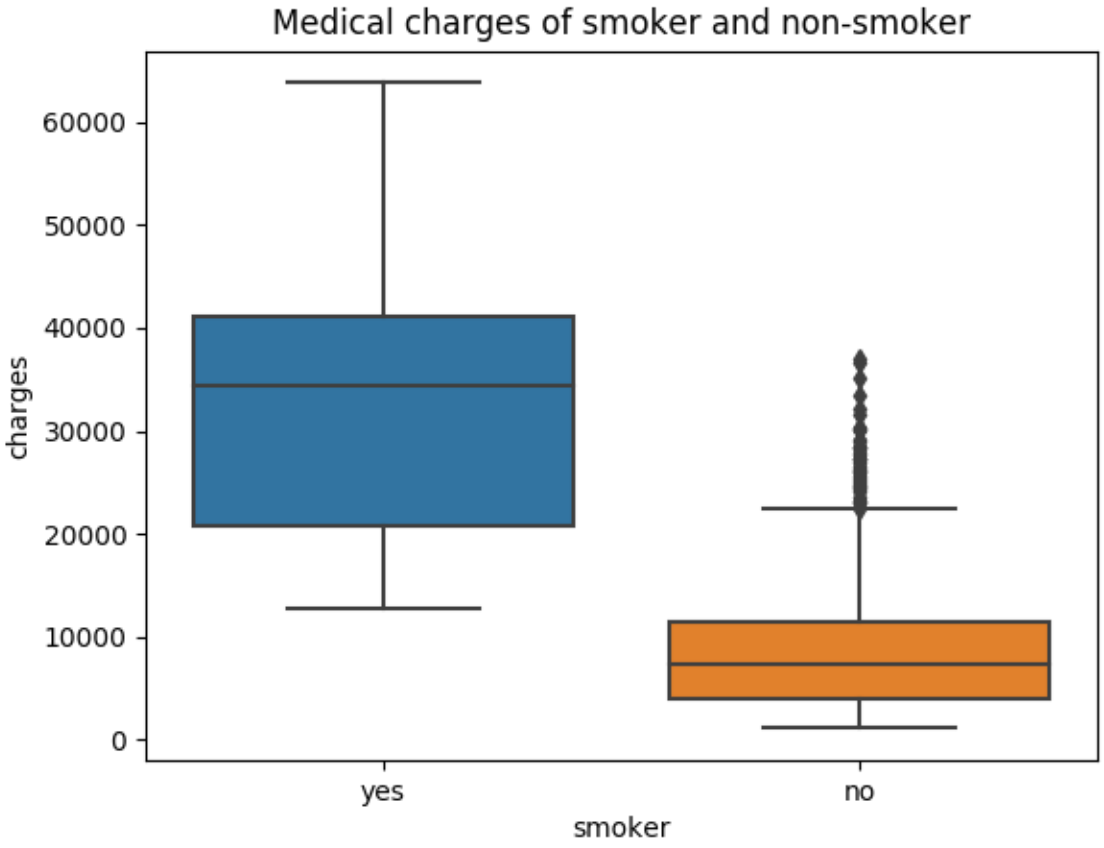
Analysis 2: Average medical cost based by BMI and Gender

- In this analysis we calculate medical charges per region i.e North, South, East, West.
- Fluctuations of medical charges for smokers and non-smokers.
- Analyze how medical charges varies with increasing BMI and age for smoker and non-smoker
- Analyze average medical cost based on body type and gender
- Medical charges analysis by gender and age groups

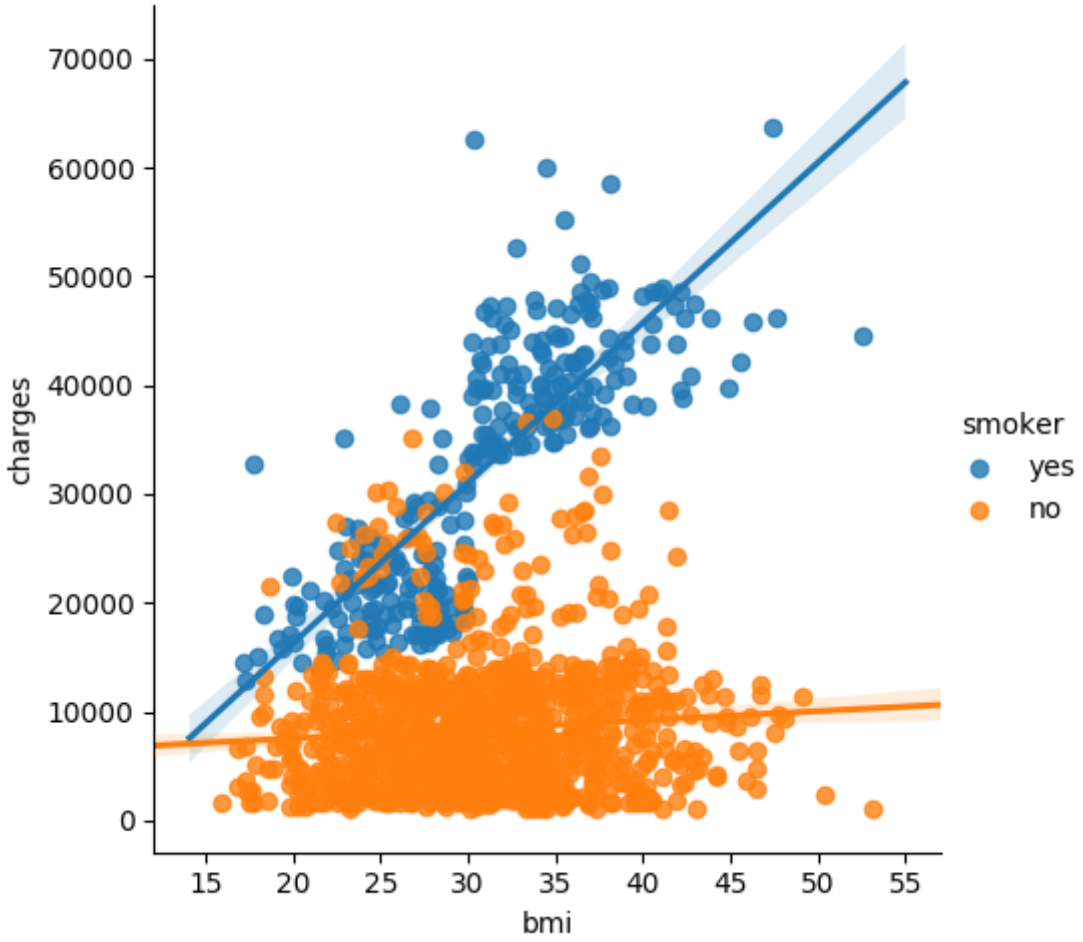
```
%scala
val insurancedf = spark.read.format("csv").option("header",
"true").option("mode",
"DROPMALFORMED").load("dbfs:/autumn_2019/pava/ProjectData/insurance.csv")
import org.apache.spark.sql.functions._
val a_insurancedf_1 = insurancedf.withColumn("BMI",
       when(col("bmi") < 18.5, "Underweight")
      .when(col("bmi") >= 18.5 && col("bmi") <= 24.9, "Normal")
      .otherwise("Overweight"))
      .groupBy("sex","BMI").agg(avg($"charges") as
"AvgMedicalCost").orderBy(desc("AvgMedicalCost"))
val a_insurancedf_2 = insurancedf.withColumn("AgeGroup",
       when(col("age") <= 20, "0-20")
      .when(col("age") > 20 && col("age") <= 40, "20-40")
      .when(col("age") > 40 && col("age") <= 60, "40-60")
      .when(col("age") > 60 && col("age") <= 80, "60-80")
      .otherwise(">80"))
      .groupBy("sex","AgeGroup").agg(avg($"charges") as
"AvgMedicalCost").orderBy(desc("AvgMedicalCost"))

insurancedf: org.apache.spark.sql.DataFrame = [age: string, sex: string ... 5 m
ore fields]
import org.apache.spark.sql.functions._
a_insurancedf_1: org.apache.spark.sql.Dataset[org.apache.spark.sql.Row] = [sex:
string, BMI: string ... 1 more field]
a_insurancedf_2: org.apache.spark.sql.Dataset[org.apache.spark.sql.Row] = [sex:
string, AgeGroup: string ... 1 more field]

import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from subprocess import check_output
df_smoker_nonsmoker=pd.read_csv('/dbfs/autumn_2019/pava/ProjectData/insurance_s
moker.csv')
sns.boxplot(x=df_smoker_nonsmoker.region,y=df_smoker_nonsmoker.charges,
data=df_smoker_nonsmoker)
plt.title("Medical Charges per region")
plt.show()
display()
```




```
sns.boxplot(x=df_smoker_nonsmoker.smoker, y=df_smoker_nonsmoker.charges,
data=df_smoker_nonsmoker)
plt.title("Medical charges of smoker and non-smoker")
display()
```

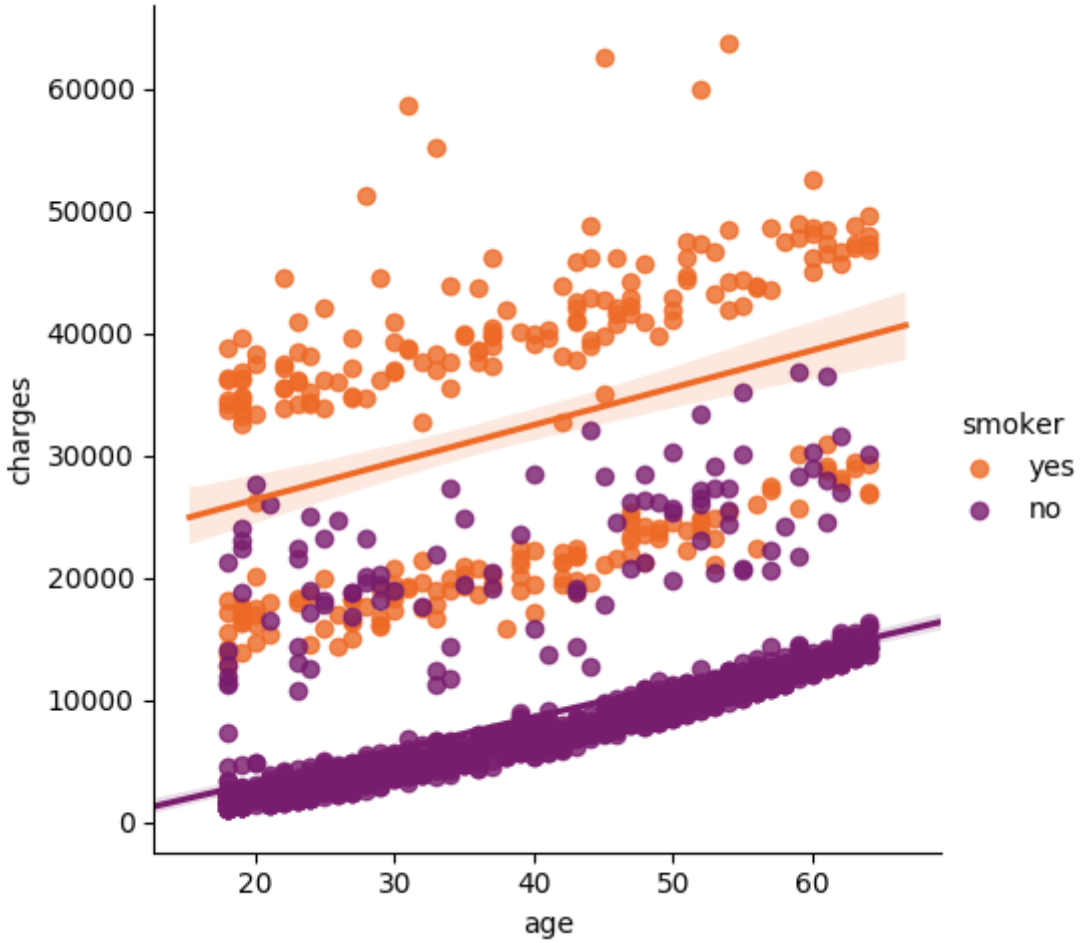



```
sns.lmplot(x="bmi", y="charges", hue='smoker',data=df_smoker_nonsmoker)
display()
```



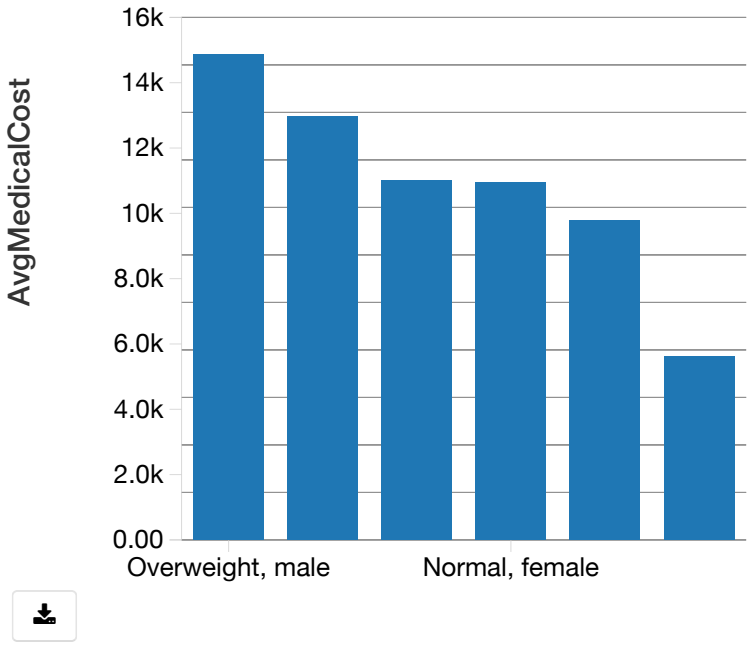
**** Medical charges increases in case of smoker with the increasing BMI. But in case of non-smokers the increasing BMI doesn't have large impact on the medical charges.***

```
sns.lmplot(x='age', y='charges',
hue='smoker',data=df_smoker_nonsmoker,palette='inferno_r')
display()
```



The cost of treatment increases with age for both smokers and non-smokers. The above graph also depicts the variation of cost between habitual and recreational smokers.

```
%scala
display(a_insurancedf_1)
```



The cost of treatment is higher for overweight males compare to overweight females, whereas the of the medical cost is higher for underweight females compare to underweight males

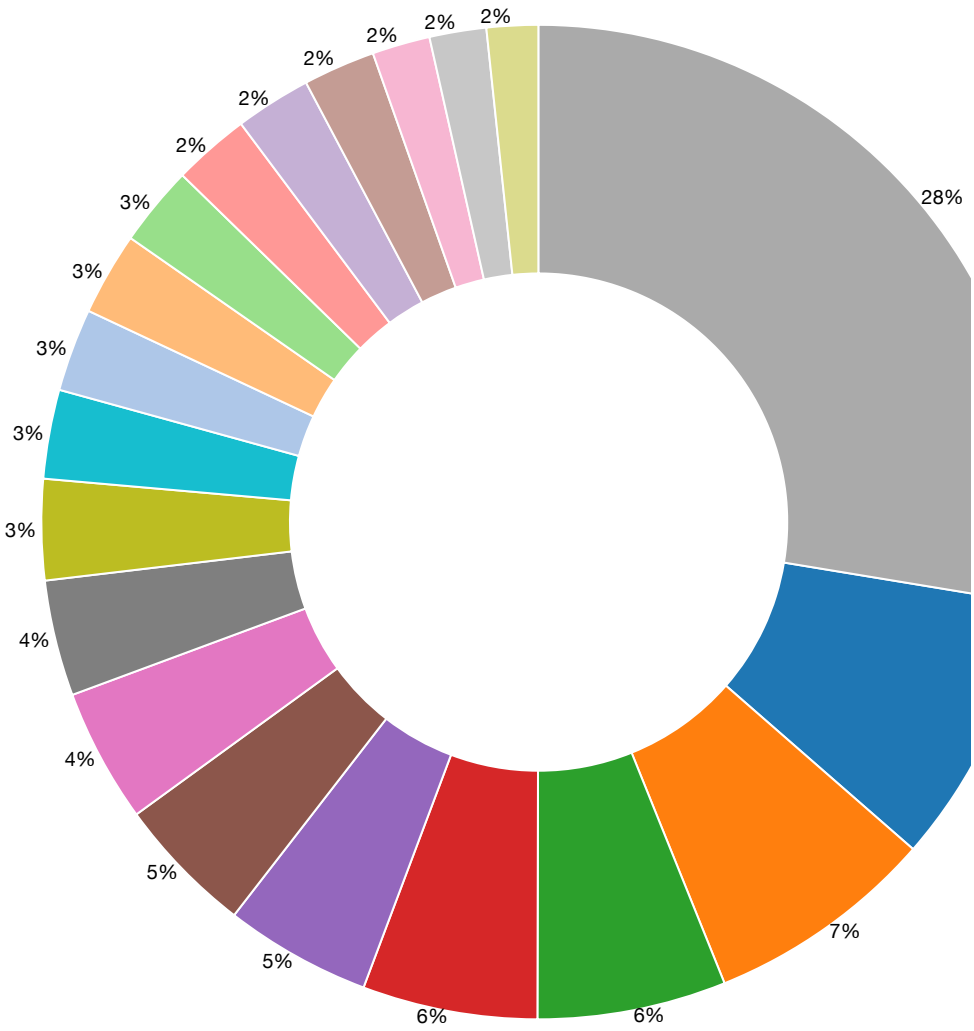
Analysis 3: Average medicare cost based on Provider Type

- In this analysis we are using Medicare Provider Utilization and Payment Data of year 2017 to do our analysis.
- This analysis focus on finding out the highest costing provider type in USA in 2017.
- Here we are also analyzing the % medicare cost of each provider type in 2017.

```
%scala
val pufdf =
spark.read.option("inferSchema","true").option("header","true").csv("dbfs:/autu
mn_2019/ektashah/ProjectDataSet/PUF_CY2017.csv")
import org.apache.spark.sql.functions._
val a_pufdf_1 = pufdf.filter($"Country Code of the Provider" ===
"US").groupBy(($"Provider Type") as "ProviderType").agg(sum($"Average Medicare
Payment Amount") as "MedicarePayment").orderBy(desc("MedicarePayment"))
```

```
pufdf: org.apache.spark.sql.DataFrame = [National Provider Identifier: int, Las
t Name/Organization Name of the Provider: string ... 24 more fields]
import org.apache.spark.sql.functions._
a_pufdf_1: org.apache.spark.sql.Dataset[org.apache.spark.sql.Row] = [ProviderTy
pe: string, MedicarePayment: double]
```

```
%scala
display(a_pufdf_1)
```



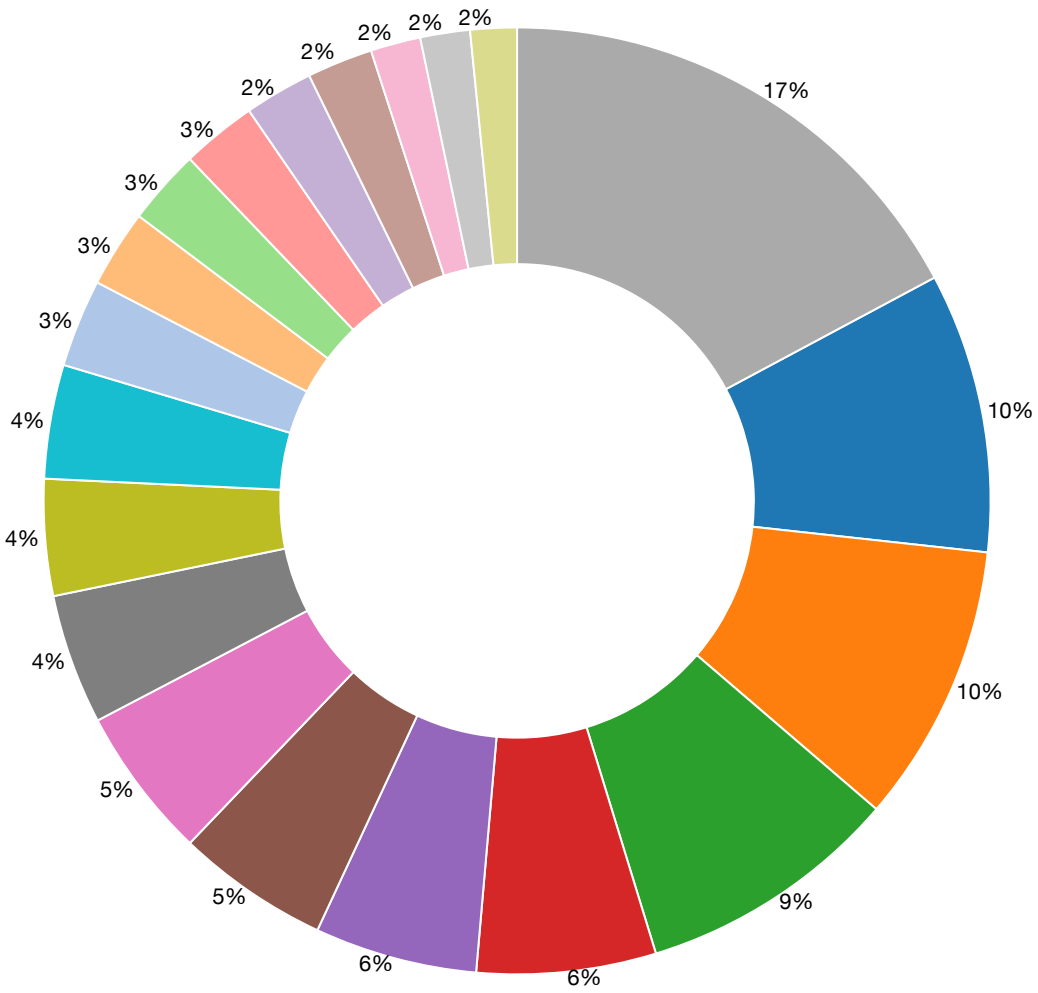
Analysis 4: Average Medicare Payment Amount Comparison based on Provider Type

- As per our last analysis Internal Medicine is the highest costing provider type, so here we are further analysing it cost in each states.

```
%scala
import org.apache.spark.sql.functions._
val a_pufdf_2 = pufdf.filter($"Country Code of the Provider" === "US" &&
$"Provider Type" === "Internal Medicine" ).groupBy("State Code of the
Provider").agg(sum($"Average Medicare Payment Amount") as
"MedicarePayment").orderBy(desc("MedicarePayment"))
```

```
import org.apache.spark.sql.functions._
a_pufdf_2: org.apache.spark.sql.Dataset[org.apache.spark.sql.Row] = [State Code
of the Provider: string, MedicarePayment: double]
```

```
%scala
display(a_pufdf_2)
```



The cost of Medicare is higher in CA and in NY for Internal Medicine Provider

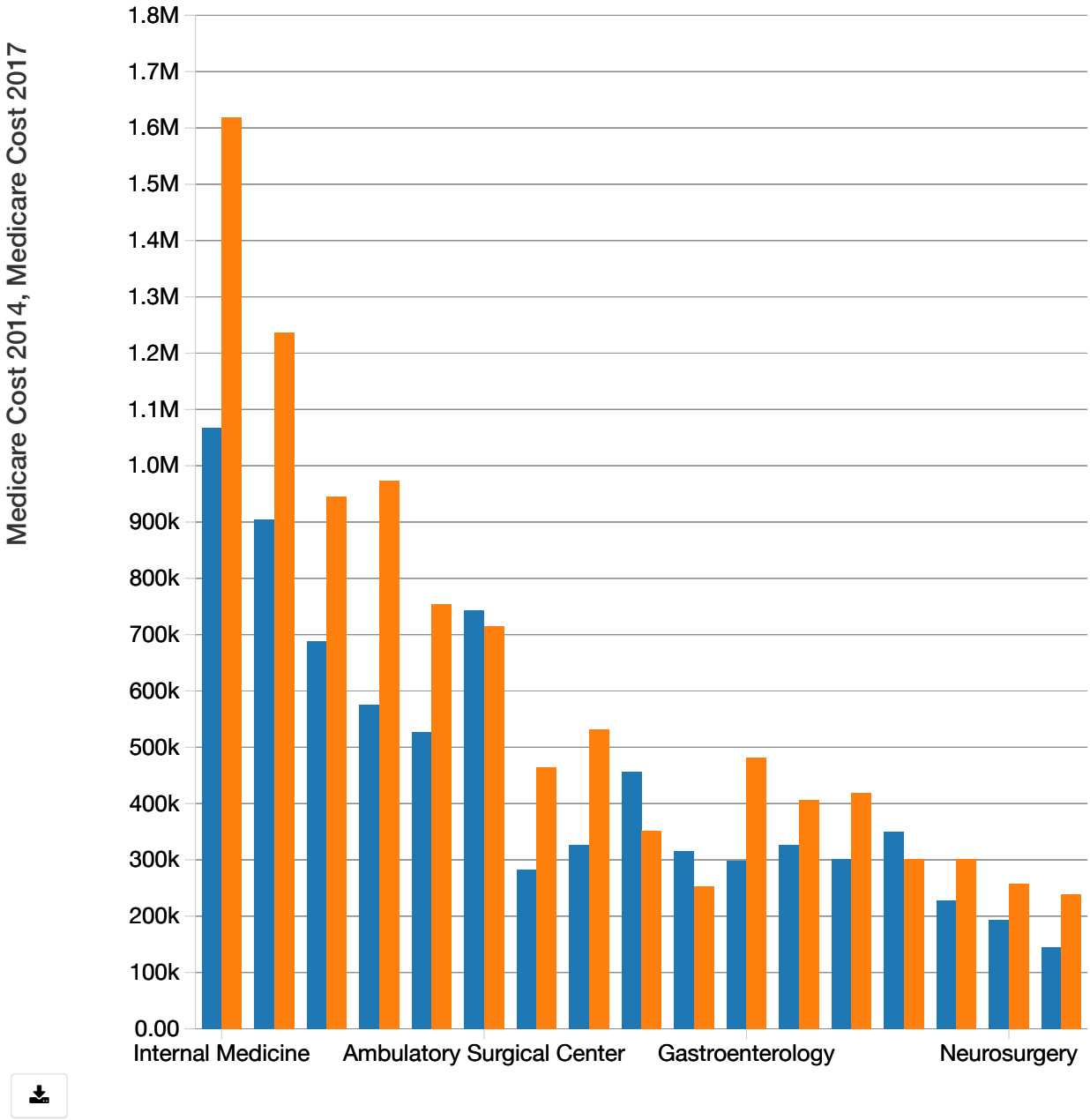
Analysis 5: Comparison between Average Medicare Cost from 2014 to 2017

- In this analysis we are comparing the Average Medicare Cost of 2014 with 2017 and analyzing the cost increment in 4 years.

```
%scala
val puf_2014 =
spark.read.option("inferSchema","true").option("header","true").csv("dbfs:/autu
mn_2019/ektashah/ProjectDataSet/PUF_CY2014.csv")
val puf_2015 =
spark.read.option("inferSchema","true").option("header","true").csv("dbfs:/autu
mn_2019/ektashah/ProjectDataSet/PUF_CY2015.csv")
val puf_2016 =
spark.read.option("inferSchema","true").option("header","true").csv("dbfs:/autu
mn_2019/ektashah/ProjectDataSet/PUF_CY2016.csv")
import org.apache.spark.sql.functions._
val a2014 = puf_2014.filter($"Country Code of the Provider" ===
"US").groupBy(($"Provider Type of the Provider") as
"ProviderType_2014").agg(sum($"Average Medicare Payment Amount") as
"MedicarePayment_2014").orderBy(desc("MedicarePayment_2014"))
val a2015 = puf_2015.filter($"Country Code of the Provider" ===
"US").groupBy(($"Provider Type") as "ProviderType_2015").agg(sum($"Average
Medicare Payment Amount") as
"MedicarePayment_2015").orderBy(desc("MedicarePayment_2015"))
val a2016 = puf_2016.filter($"Country Code of the Provider" ===
"US").groupBy(($"Provider Type") as "ProviderType_2016").agg(sum($"Average
Medicare Payment Amount") as
"MedicarePayment_2016").orderBy(desc("MedicarePayment_2016"))
val a2017 = pufdf.filter($"Country Code of the Provider" ===
"US").groupBy(($"Provider Type") as "ProviderType_2017").agg(sum($"Average
Medicare Payment Amount") as
"MedicarePayment_2017").orderBy(desc("MedicarePayment_2017"))
val join2016  =  a2017.join(a2016, $"ProviderType_2017" ===
$"ProviderType_2016").select($"ProviderType_2017", $"MedicarePayment_2017",
$"MedicarePayment_2016")
val join2015  =  join2016.join(a2015, $"ProviderType_2017" ===
$"ProviderType_2015").select($"ProviderType_2017", $"MedicarePayment_2017",
$"MedicarePayment_2016", $"MedicarePayment_2015")
val join2014  =  join2015.join(a2014, $"ProviderType_2017" ===
$"ProviderType_2014").select($"ProviderType_2017", $"MedicarePayment_2017",
$"MedicarePayment_2016",$"MedicarePayment_2015",$"MedicarePayment_2014")


puf_2014: org.apache.spark.sql.DataFrame = [National Provider Identifier: int,
Last Name/Organization Name of the Provider: string ... 24 more fields]
puf_2015: org.apache.spark.sql.DataFrame = [National Provider Identifier: int,
Last Name/Organization Name of the Provider: string ... 24 more fields]
puf_2016: org.apache.spark.sql.DataFrame = [National Provider Identifier: int,
Last Name/Organization Name of the Provider: string ... 24 more fields]
import org.apache.spark.sql.functions._
a2014: org.apache.spark.sql.Dataset[org.apache.spark.sql.Row] = [ProviderType_2
014: string, MedicarePayment_2014: double]
```

```
a2015: org.apache.spark.sql.Dataset[org.apache.spark.sql.Row] = [ProviderType_2
015: string, MedicarePayment_2015: double]
a2016: org.apache.spark.sql.Dataset[org.apache.spark.sql.Row] = [ProviderType_2
016: string, MedicarePayment_2016: double]
a2017: org.apache.spark.sql.Dataset[org.apache.spark.sql.Row] = [ProviderType_2
017: string, MedicarePayment_2017: double]
join2016: org.apache.spark.sql.DataFrame = [ProviderType_2017: string, Medicare
Payment_2017: double ... 1 more field]
join2015: org.apache.spark.sql.DataFrame = [ProviderType_2017: string, Medicare
Payment_2017: double ... 2 more fields]
join2014: org.apache.spark.sql.DataFrame = [ProviderType_2017: string, Medicare
Payment_2017: double ... 3 more fields]
```

Show code



The overall Medical Cost has increased for all the Provider Types from 2014 to 2017

```
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns
import numpy as np
df_cost=pd.read_csv('/dbfs/autumn_2019/pava/ProjectData/IPPS_Data_Clean.csv')
df_cost.head()
```

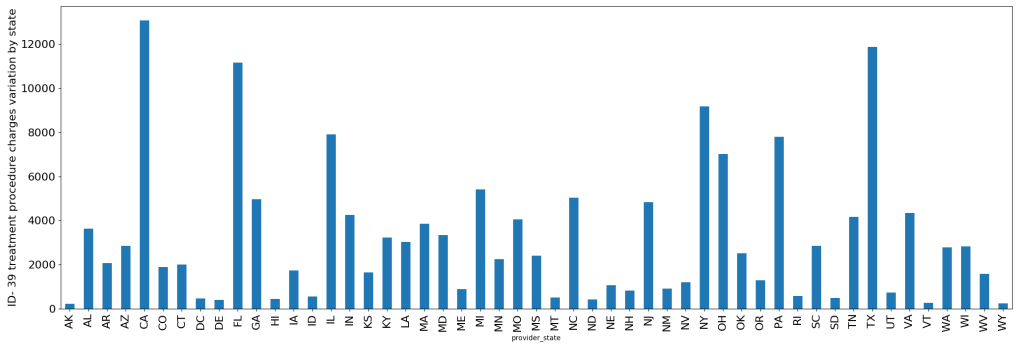
Out[2]:

	drg_id	drg_definition	provider_id	provider_name	provider_street_address	provider_city	pr
0	39	039 - EXTRACRANIAL PROCEDURES W/O CC/MCC	10001	SOUTHEAST ALABAMA MEDICAL CENTER	1108 ROSS CLARK CIRCLE	DOTHAN	
1	39	039 - EXTRACRANIAL PROCEDURES W/O CC/MCC	10005	MARSHALL MEDICAL CENTER SOUTH	2505 U S HIGHWAY 431 NORTH	BOAZ	
2	39	039 - EXTRACRANIAL PROCEDURES W/O CC/MCC	10006	ELIZA COFFEE MEMORIAL HOSPITAL	205 MARENGO STREET	FLORENCE	
3	39	039 - EXTRACRANIAL PROCEDURES W/O CC/MCC	10011	ST VINCENT'S EAST	50 MEDICAL PARK EAST DRIVE	BIRMINGHAM	
4	39	039 - EXTRACRANIAL PROCEDURES W/O CC/MCC	10016	SHELBY BAPTIST MEDICAL CENTER	1000 FIRST STREET NORTH	ALABASTER	

- Billed charges vary from hopital to hospital for the same procedure.

```
nationalmedian = pd.DataFrame(df_cost.groupby('drg_id',sort=False)
['average_covered_charges','average_total_payments','average_medicare_payments'
].median()).reset_index()
nationalmedian = nationalmedian.rename(columns=
{'average_covered_charges':'median_covered_charges',
'average_total_payments':'median_total_payments','average_medicare_payments':'m
edian_medicare_payments' })
```

```
costoftreatmentperstate=df_cost.groupby(['drg_id','provider_state']).size()
costoftreatmentperstate
=costoftreatmentperstate.unstack('provider_state').fillna(0)
costoftreatmentperstate.sum().plot(ylim=0,kind='bar',figsize=
(25,8),fontsize=16);
plt.ylabel('ID- 39 treatment procedure charges variation by state',fontsize=16)
display()
```

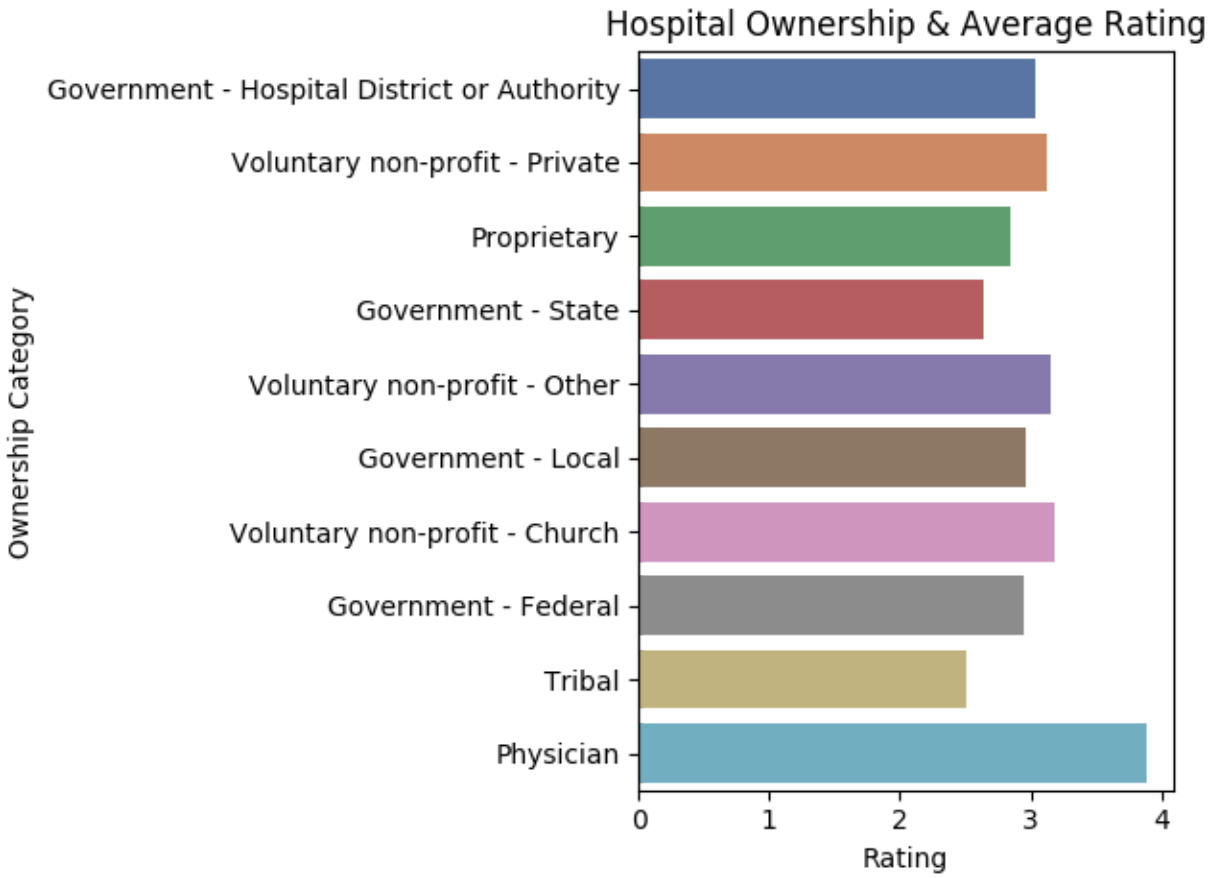


For the same treatment, variation of medical charges with respect to state

Analysis 6: Hospital rating

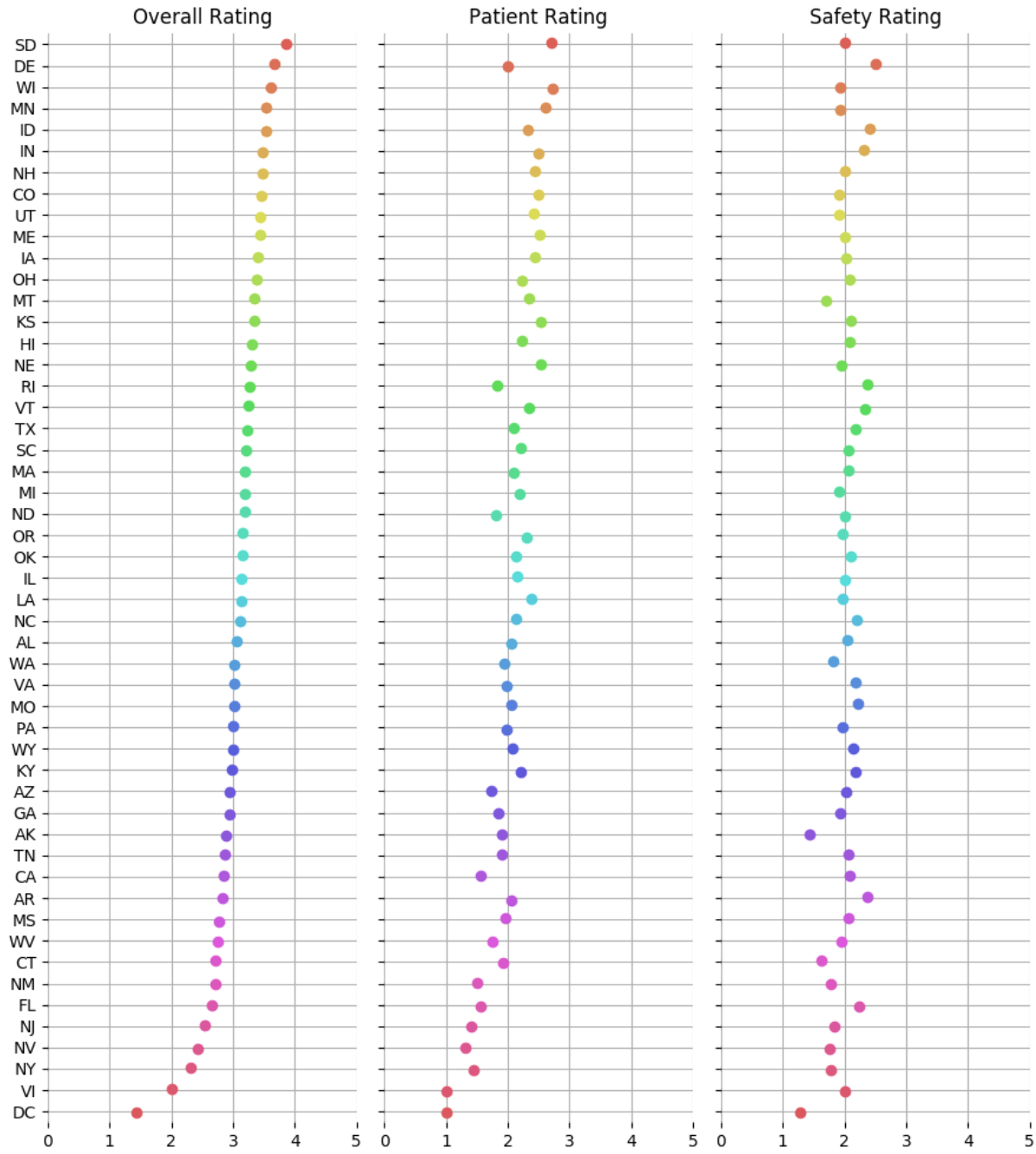
- Hospital ownership (Government, private, non-profit) and their average rating
- The average rating is based on hospital overall rating, saftey of care national rating, patient experience.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.axes_grid1 import make_axes_locatable
import seaborn as sns
import scipy as sp
df_hospitalrating=pd.read_csv("/dbfs/autumn_2019/pava/ProjectData/HospInfo.csv"
)
df_ratings=df_hospitalrating.drop(['Address','ZIP Code','Phone
Number','Emergency Services',
'Patient experience national comparison footnote','Efficient use of medical
imaging national comparison footnote',
'Hospital overall rating footnote','Mortality national comparison
footnote','Mortality national comparison footnote',
'Safety of care national comparison footnote','Readmission national comparison
footnote',
'Patient experience national comparison footnote','Effectiveness of care
national comparison footnote',
'Timeliness of care national comparison footnote','Timeliness of care national
comparison footnote',
'Efficient use of medical imaging national comparison footnote','Meets criteria
for meaningful use of EHRs',
'Provider ID'], axis=1)
df1=df_ratings.replace('Below the National average',1)
df2=df1.replace('Same as the National average',2)
df3=df2.replace('Above the National average',3)
df3['Hospital overall rating'] = df3['Hospital overall
rating'].convert_objects(convert_numeric=True)
sns.barplot(x='Hospital overall rating', y='Hospital Ownership',ci=0, data=df3,
palette="deep")
plt.title('Hospital Ownership & Average Rating')
plt.xlabel('Rating')
plt.ylabel('Ownership Category')
plt.tight_layout()
display()
```


Hospital Ownership & Average Rating
Government - Hospital District or Authority
Voluntary non-profit - Private
Proprietary
Government - State
Voluntary non-profit - Other
Government - Local
Voluntary non-profit - Church
Government - Federal
Tribal
Physician
Ownership Category
0
1
2
3
4
Rating

```
df1=df_ratings.replace('Below the National average',1)
df2=df1.replace('Same as the National average',2)
df3=df2.replace('Above the National average',3)


# Convert types to numeric:
df3['Mortality national comparison'] = df3['Mortality national
comparison'].convert_objects(convert_numeric=True)
df3['Safety of care national comparison'] = df3['Safety of care national
comparison'].convert_objects(convert_numeric=True)
df3['Readmission national comparison'] = df3['Readmission national
comparison'].convert_objects(convert_numeric=True)
df3['Patient experience national comparison'] = df3['Patient experience
national comparison'].convert_objects(convert_numeric=True)
df3['Mortality national comparison'] = df3['Mortality national
comparison'].convert_objects(convert_numeric=True)
df3['Hospital overall rating'] = df3['Hospital overall
rating'].convert_objects(convert_numeric=True)

df5=df3.pivot_table(index=['State'], values = ['Hospital overall
rating','Safety of care national comparison',
                    'Patient experience national comparison'], aggfunc='mean')
df5=pd.DataFrame(df5)
df5=df5.reset_index()
df5=df5.sort_values("Hospital overall rating", ascending = False).dropna()
#Ratings Correlation
g = sns.PairGrid(df5,
                 x_vars=['Hospital overall rating','Patient experience national
comparison','Safety of care national comparison']
                 , y_vars=["State"],
                 size=10, aspect=.3)

# Draw a dot plot using the stripplot function
g.map(sns.stripplot, size=7, orient="h", palette="hls")

# Use the same x axis limits on all columns and add better labels
g.set(xlim=(0, 5), xlabel="", ylabel="")

# Use semantically meaningful titles for the columns
titles = ['Overall Rating','Patient Rating','Safety Rating']

for ax, title in zip(g.axes.flat, titles):

    # Set a different title for each axes
    ax.set(title=title)

    # Make the grid horizontal instead of vertical
```

```
    ax.xaxis.grid(True)
    ax.yaxis.grid(True)
plt.tight_layout()
sns.despine(left=True, bottom=True)
display()
```



Analysis 7: Hospital Readmission

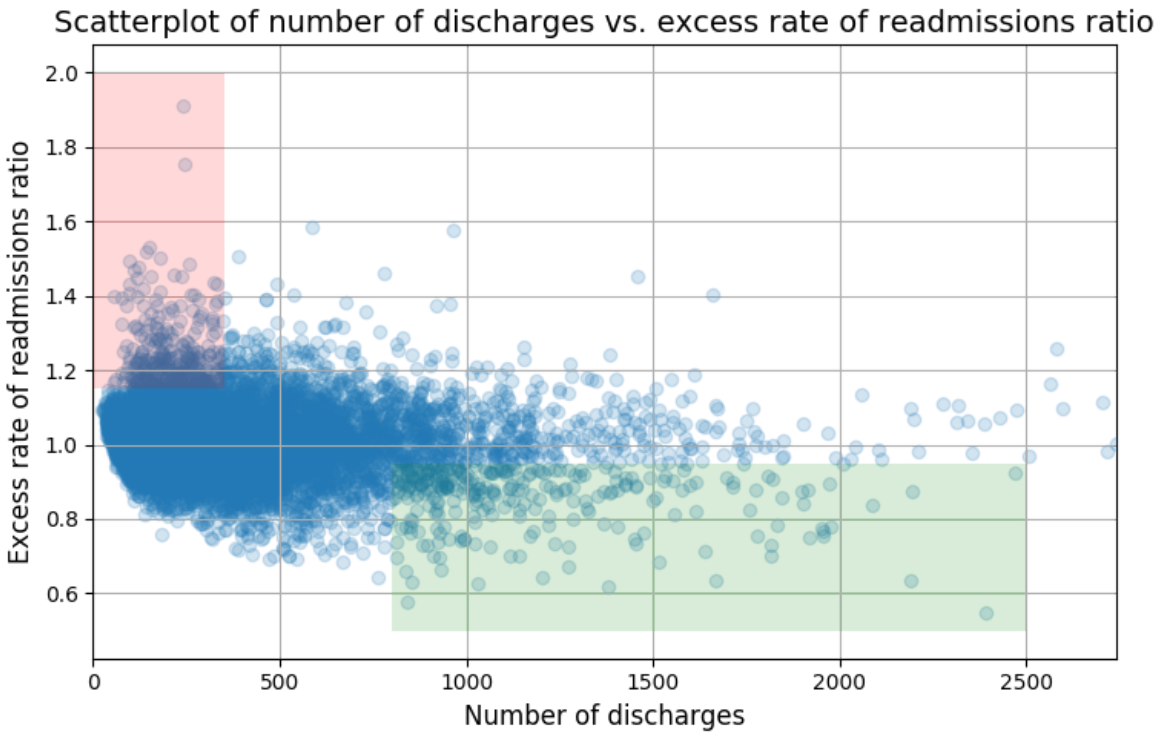
- In this analysis we try to find if there is any relation between number of discharges and readmission ratio.
- Readmission ratio is the ratio of predicted rate/Expected rate.
- predicated rate - 30 day readmission for heart attack (AMI), heart failure (HF), pneumonia, chronic obstructive pulmonary disease (COPD), hip/knee replacement (THA/TKA), and coronary artery bypass graft surgery (CABG).

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.axes_grid1 import make_axes_locatable
import seaborn as sns
import scipy as sp
df_readmission=pd.read_csv("/dbfs/autumn_2019/pava/ProjectData/cms_readmissions
.csv")
df_readmission.head()
```

Out[1]:

	Hospital Name	Provider Number	State	Measure Name	Number of Discharges	Footnote	Excess Readmission Ratio	Predicted Readmission Rate	Re
0	FROEDTERT MEMORIAL LUTHERAN HOSPITAL	520177	WI	READM-30-HIP-KNEE-HRRP	242	NaN	1.9095	10.8	
1	PROVIDENCE HOSPITAL	90006	DC	READM-30-HIP-KNEE-HRRP	247	NaN	1.7521	9.2	
2	BEAUFORT COUNTY MEMORIAL HOSPITAL	420067	SC	READM-30-HIP-KNEE-HRRP	586	NaN	1.5836	7.6	

Show code



- Rate of readmission is trending down with increase in number of discharges.
- With lower number of discharges there is excess rate of readmission.
- With higher number of discharges lower rate of readmissions.

Statistics:

- In hospitals with discharges <100, mean excess readmission rate is 1.023 and 63% have excess readmission rate greater than 1.
- In hospitals with discharges>1000, mean excess readmission rate is 0.978 and 44% have excess readmission rate greater than 1.

Conclusion:

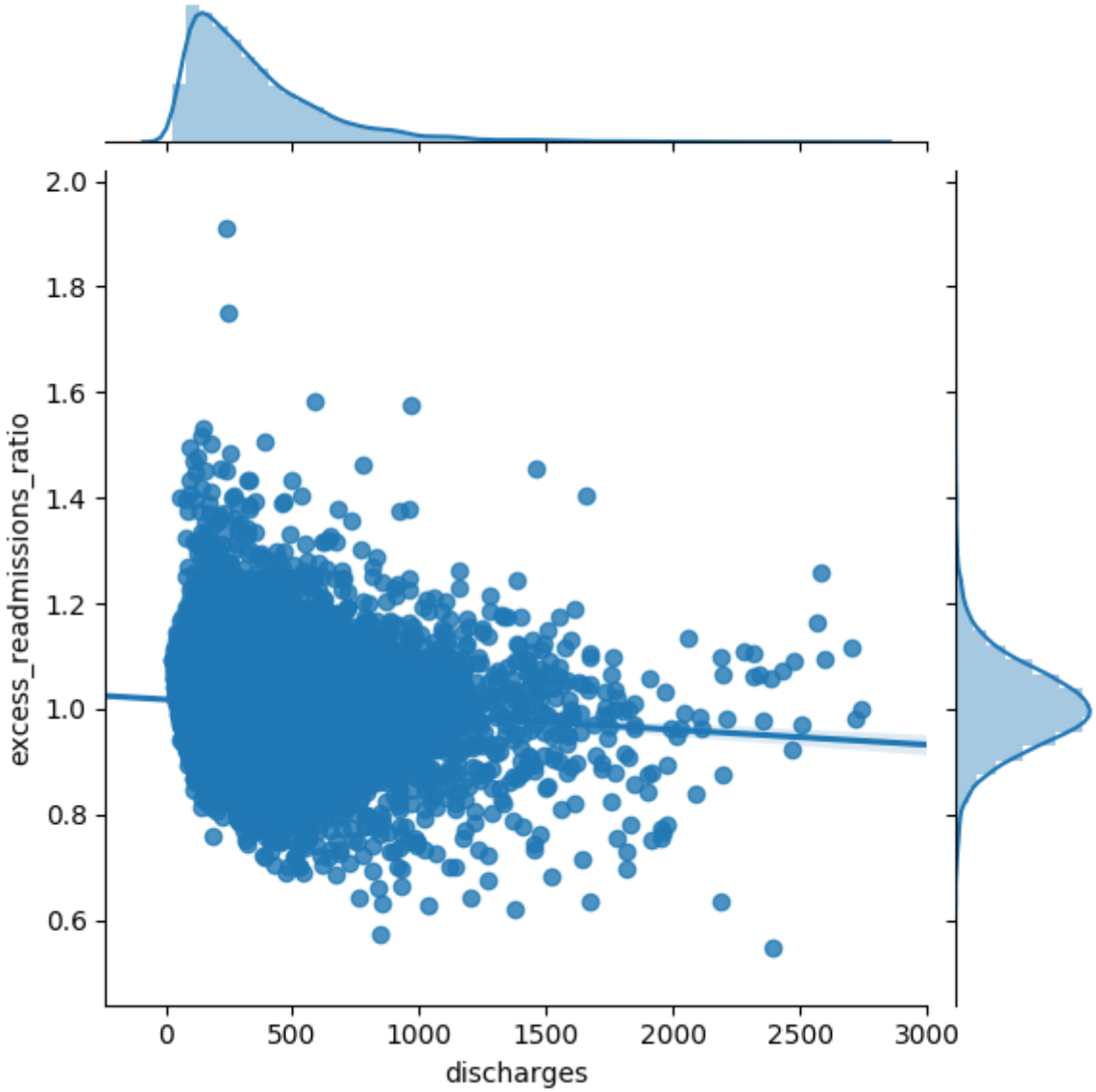
- There is a significant correlation between hospital capacity and readmission rates.
- Smaller hospitals/facilities may be lacking necessary resources to ensure quality care this may lead to readmission rate greater than 1.

```
x=pd.DataFrame(x)
y=pd.DataFrame(y)
df = pd.concat([x,y], axis=1)
df.columns = ['discharges', 'excess_readmissions_ratio']
df.head()
```

Out[3]:

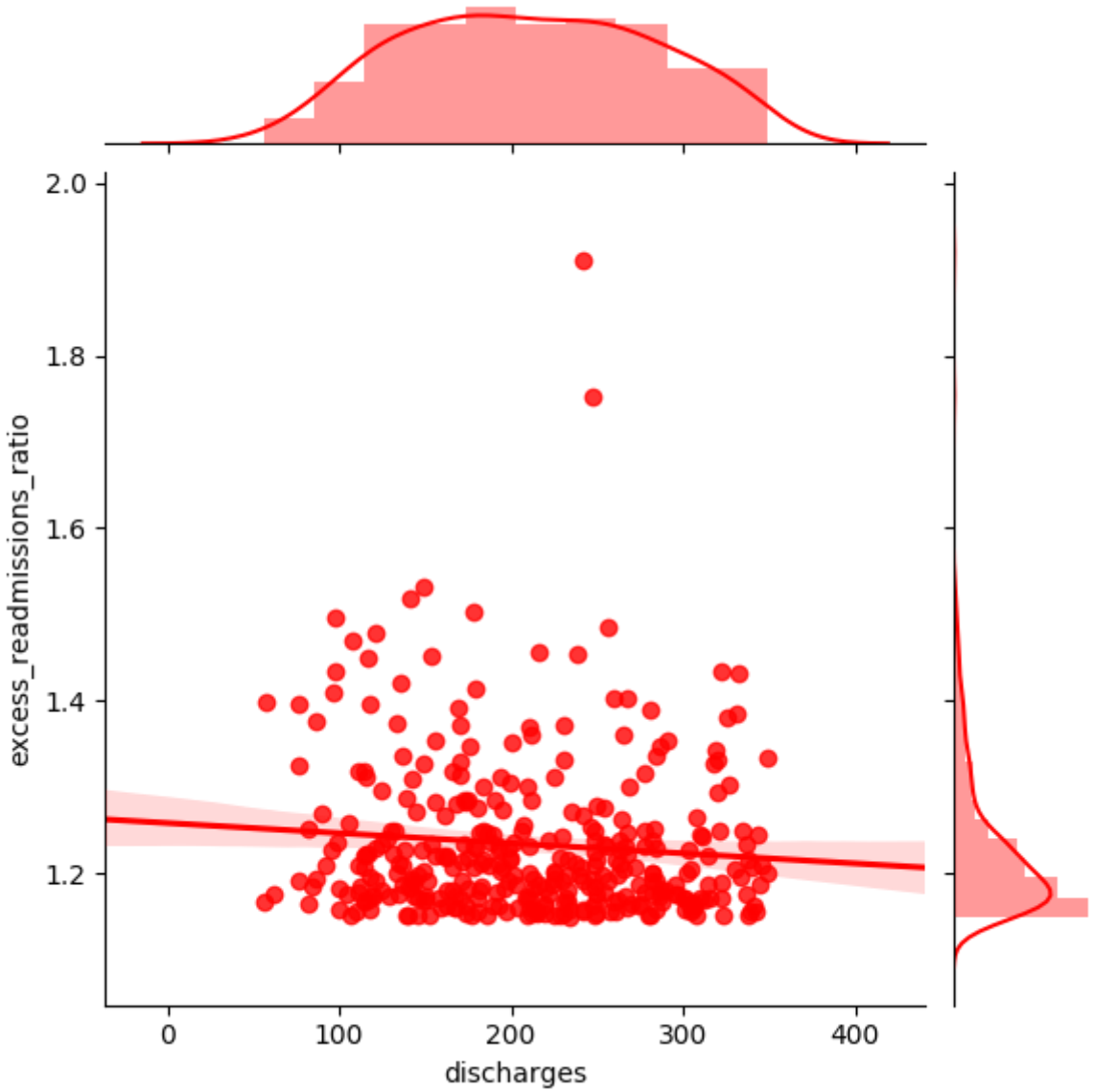
	discharges	excess_readmissions_ratio
0	25	1.0914
1	27	1.0961
2	28	1.0934
3	29	1.0908
4	30	1.1123

Show code



Overall, rate of readmission is slightly trending down with the increasing number of discharges.

Show code

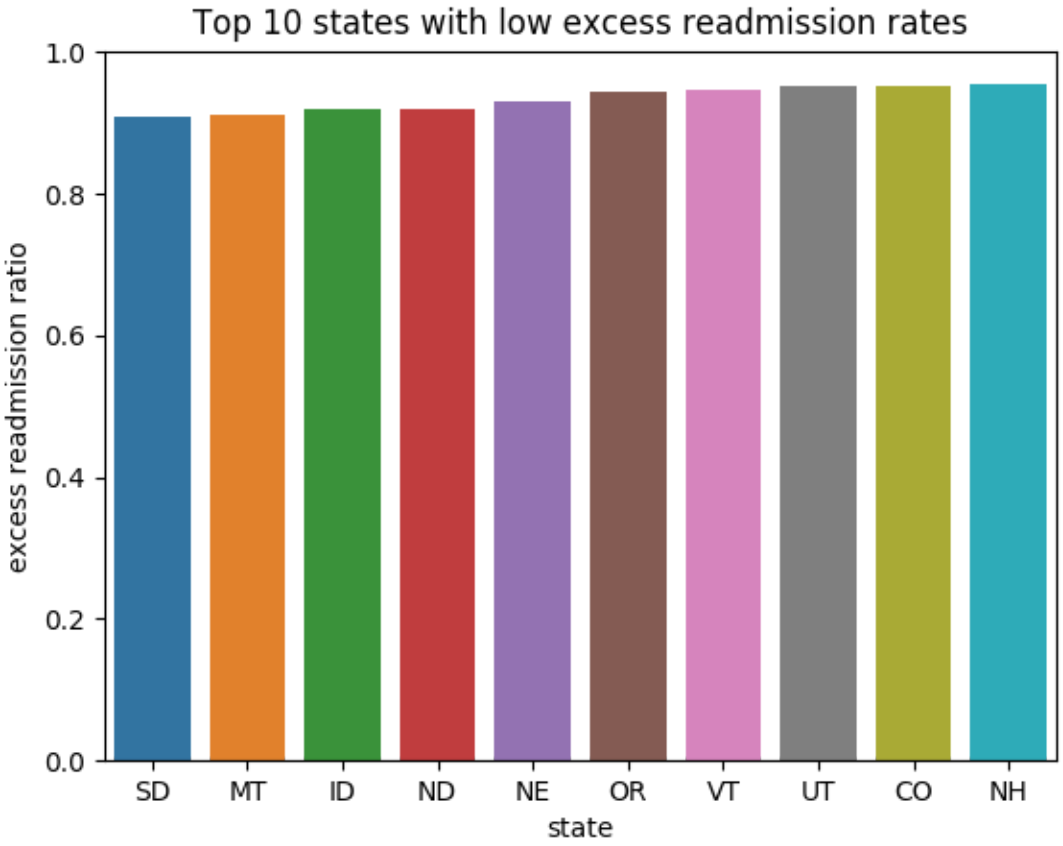


With lower number of discharges, there is a greater incidence of excess rate of readmissions

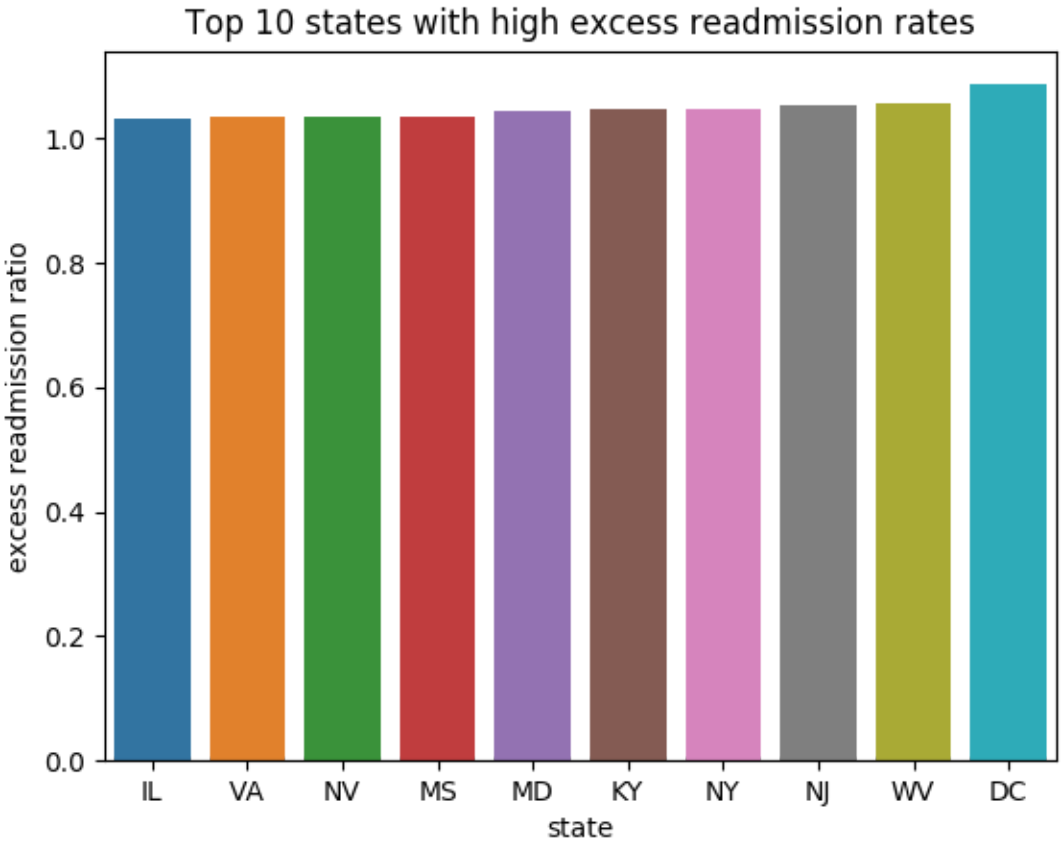
Readmissions by state:

- Top 10 states with low and high excess readmission rates

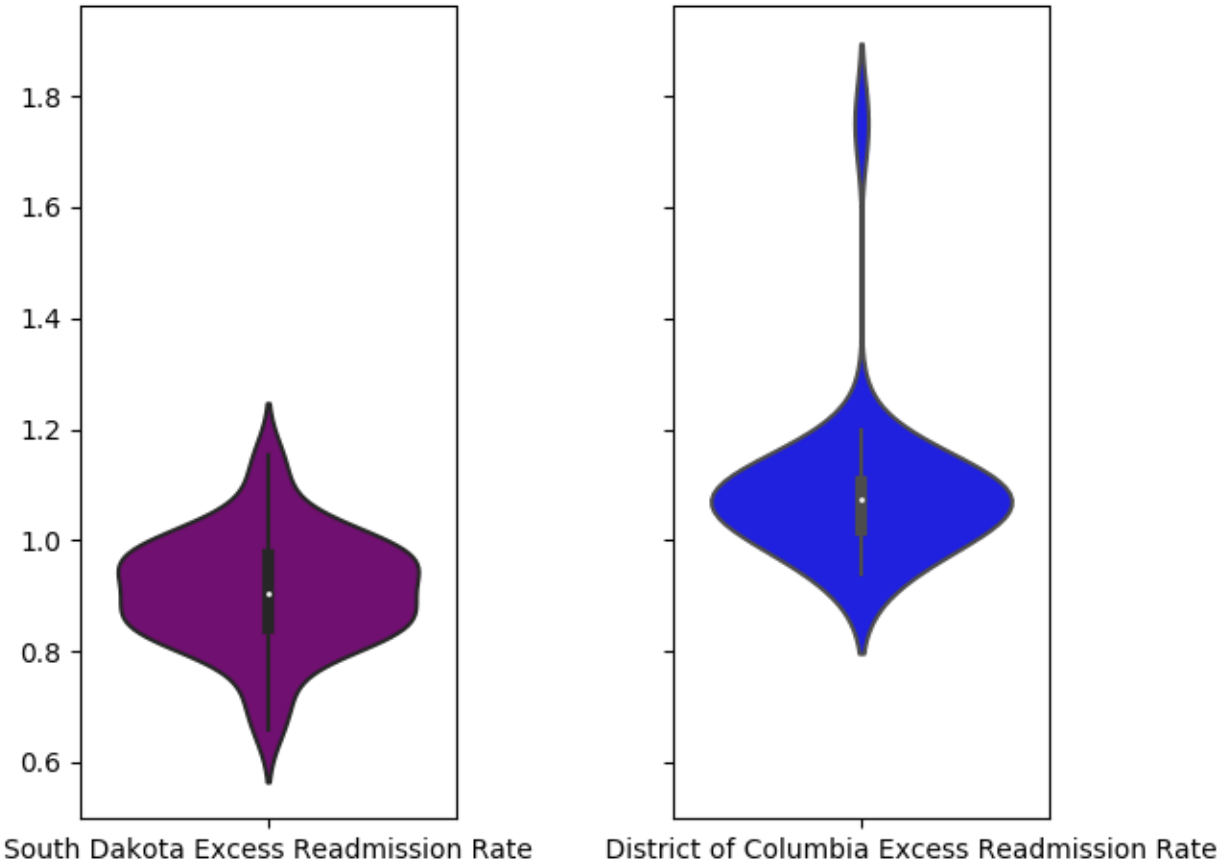
Show code


Top 10 states with low excess readmission rates
excess readmission ratio
1.0
0.8
0.6
0.4
0.2
0.0
SD
MT
ID
ND
NE
OR
VT
UT
CO
NH
state

Show code


Top 10 states with high excess readmission rates
excess readmission ratio
0.0
0.2
0.4
0.6
0.8
1.0
IL
VA
NV
MS
MD
KY
NY
NJ
WV
DC
state

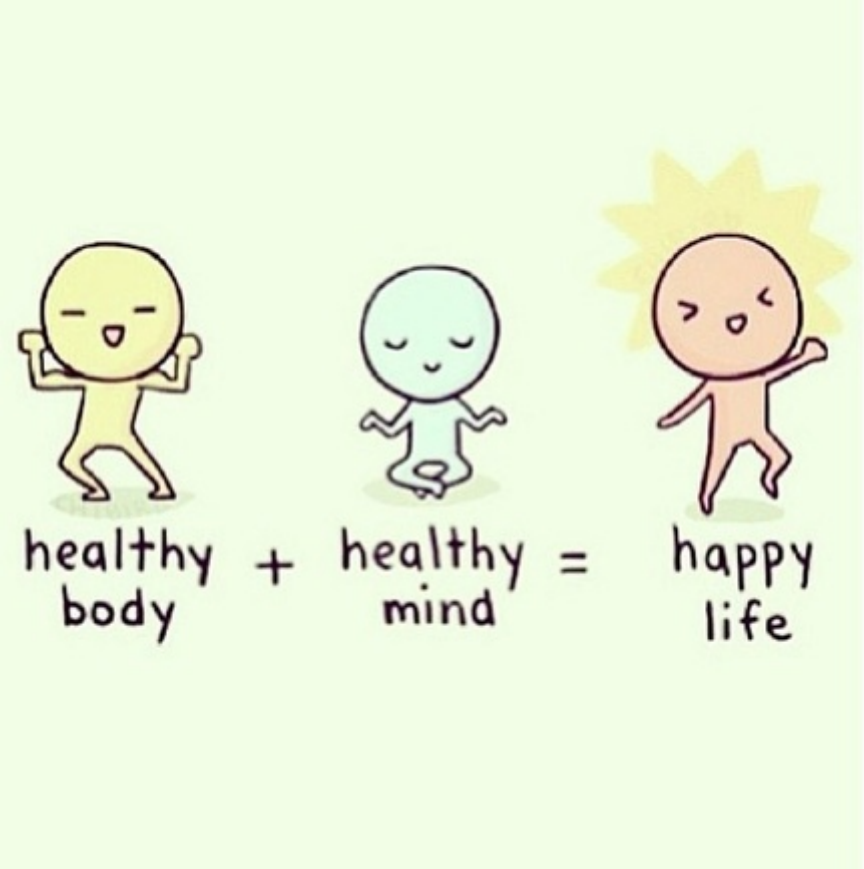
Show code



- Violin plot is a combination of box plot and density plot.
- From the above analysis South Dakota(SD) had lower readmission ratio and District of Columbia (DC) had the greatest readmission ratio.
- From the violin plot it can be seen that the SD mean readmission ratio is around 0.9 and the distribution is uniform.
- District of Columbia mean readmission ratio is 1.1 and their is a slight fluctuation in the distribution of data.

Result:

- If people cultivate healthy life style then medical cost reduces :)
- The south east region has higher fluctuations in medical cost.
- For smokers the medical cost increases indisputably with BMI and age.
- Its better to prefer hospital with higher rate of discharge and lower readmission ratio.


healthy body
+
healthy mind
=
happy life