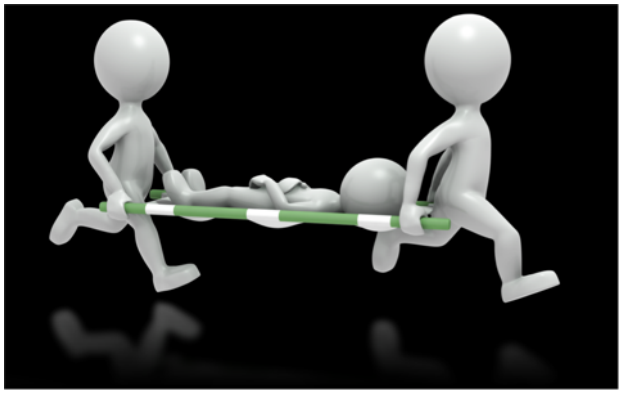
The Center for Medicare and Medicaid Services
- HealthCare Fraud Detection -

HealthCare Fraud Problems



Healthcare Fraud Problems

- 3%-10% of the USA's health care spending
- In Michigan alone, 51,322 cases have been opened and saved $402 million
- Medicare's contractors process 4.5 million claims a day

The problem...

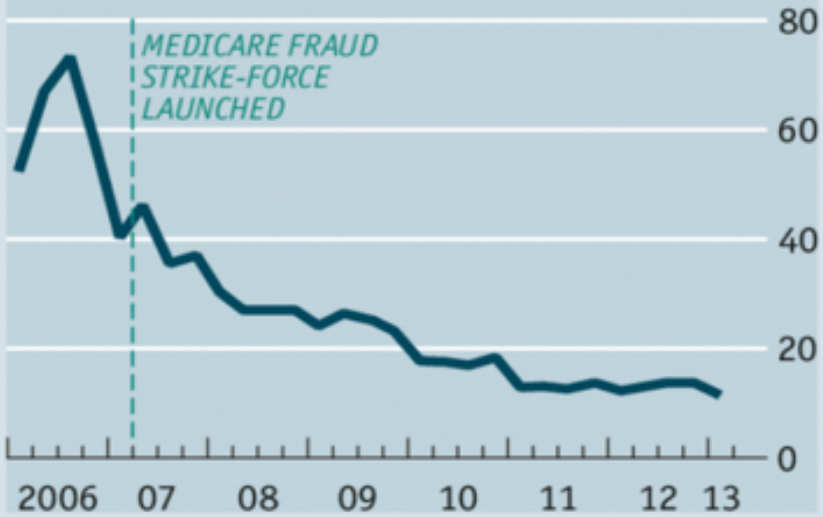
Healthcare spending lost to fraud and abuse

	Low	Mid-point	High
2011 estimates:	$82bn	$177bn	$272bn
% of total healthcare spending:	*3.1*	*6.7*	*10.2*

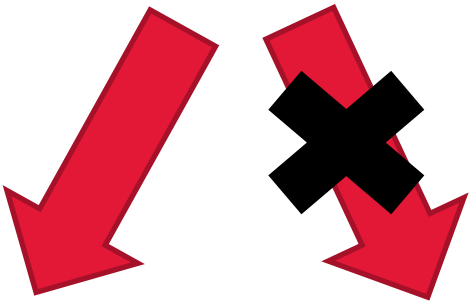
...and a partial solution...

Medicare spending on medical equipment, $m

(Miami-Miami Beach-Kendall, Florida)

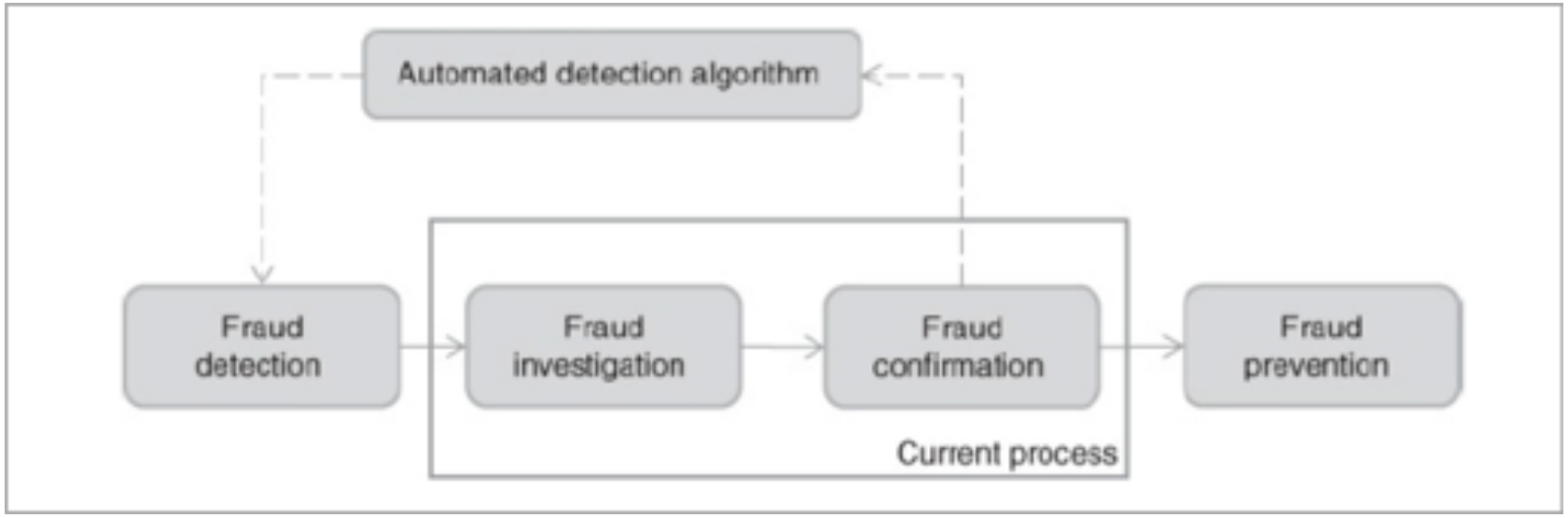


Sources: D.M. Berwick and A.D. Hackbarth, Eliminating Waste in US Health Care; Department of Health and Human Services

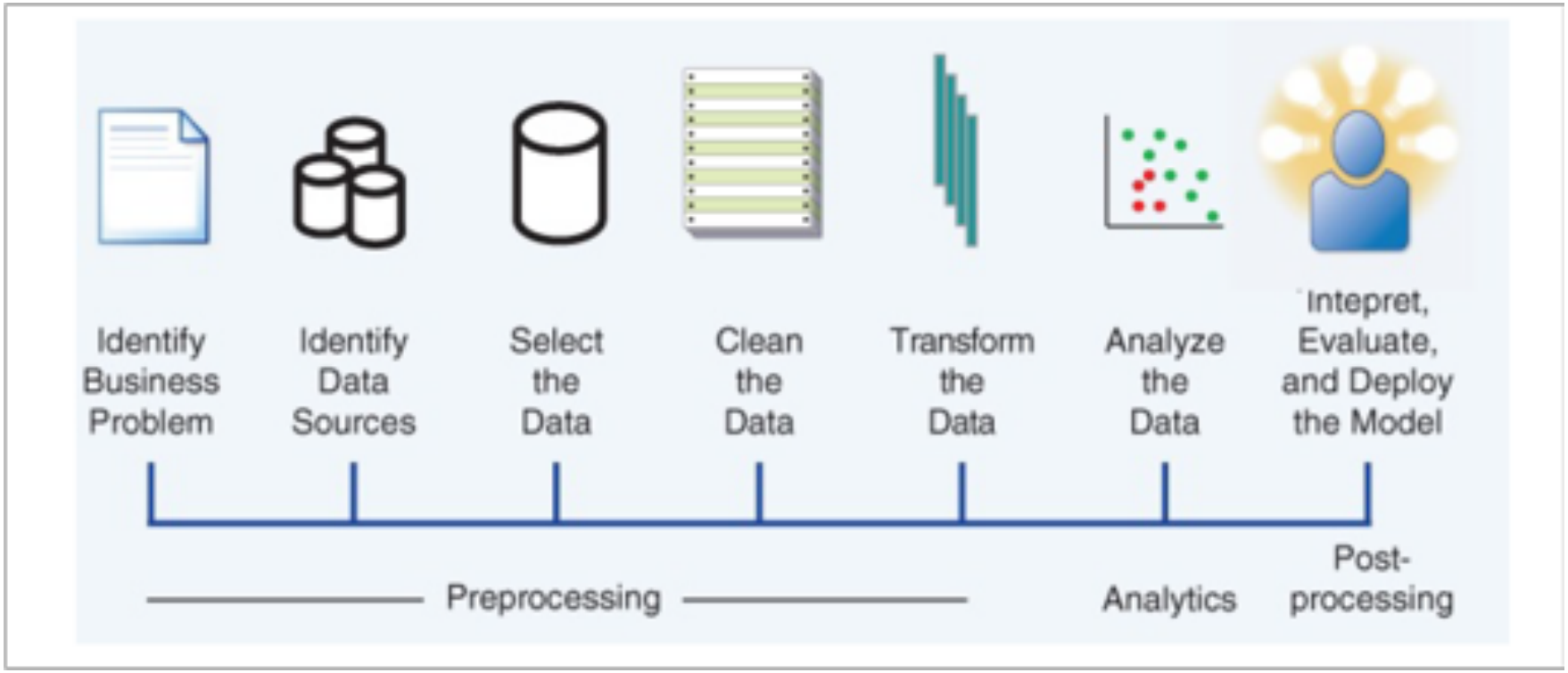


Data-Driven based Fraud Detection

- Precision
- Operational efficiency
- Cost efficiency

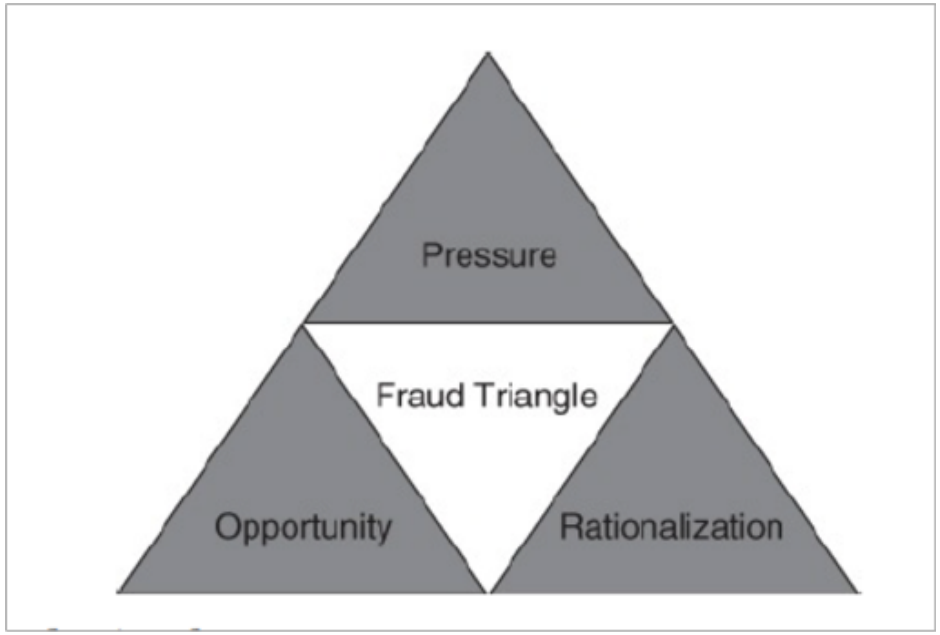


The Fraud Detection Analytics Process Model



Key Characteristics of Successful Fraud Analytics Models

- Statistical accuracy
- Interpretability
- Operational efficiency
- Economical Cost
- Regulatory Compliance



Data Collection, Sampling, and Preprocessing

- **Types of Data Sources**
  CMS Part D data; Payment Data; Exclusive Data
  Opioid Drugs Data; Crime Data

- **Types of Data Elements**
  Numeric Data : values
  Categorical data : Location, Drugs names, npi …

- **Visual Data Exploration and Exploratory Statistical Analysis**

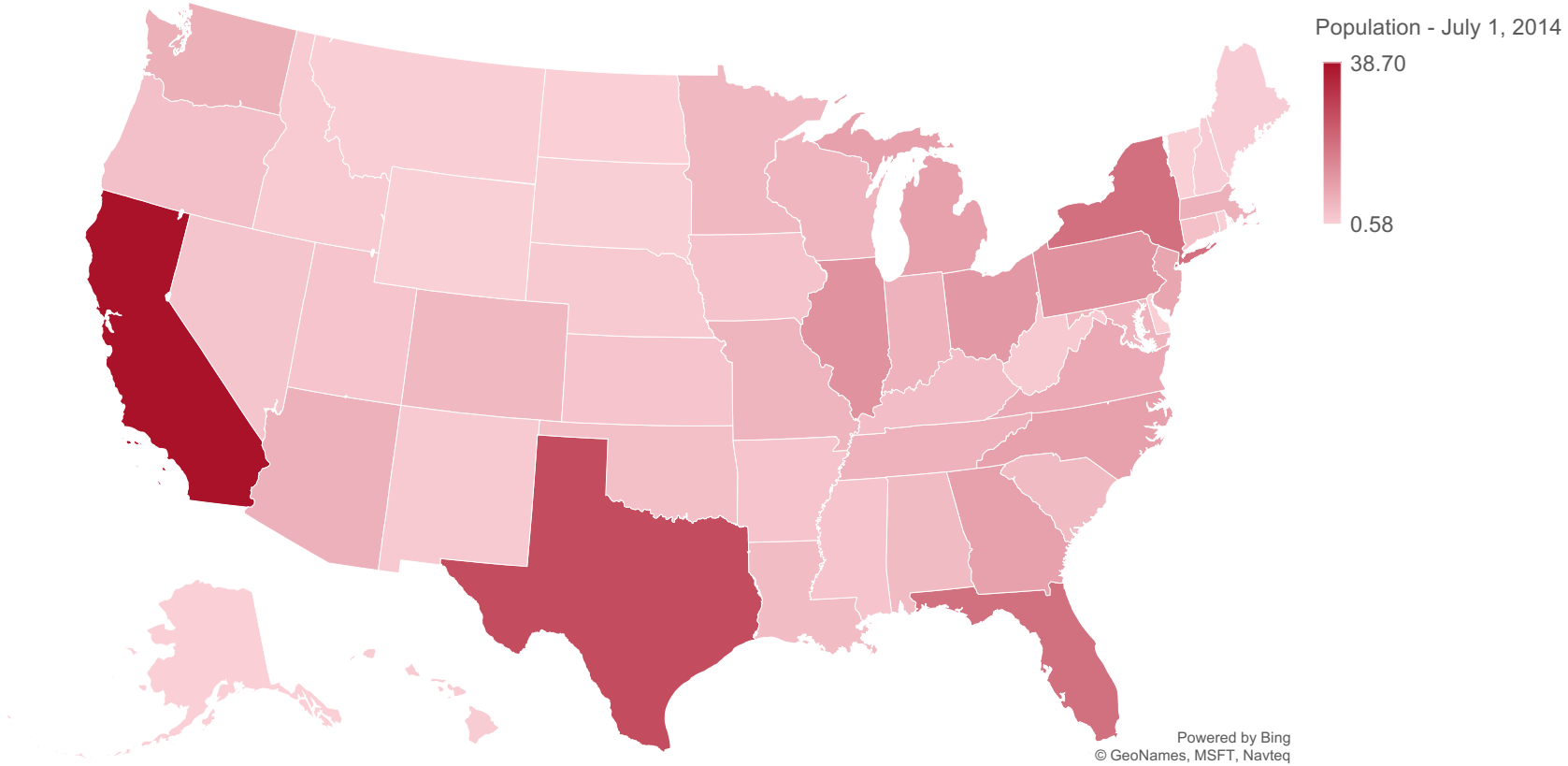
- **How to Handle the huge Dataset**
  API – Google Big Query /Socrata API
  Big Data Analytics – Hadoop/Pig/PySpark
  Python Pandas

Data Sampling and Pre-processing

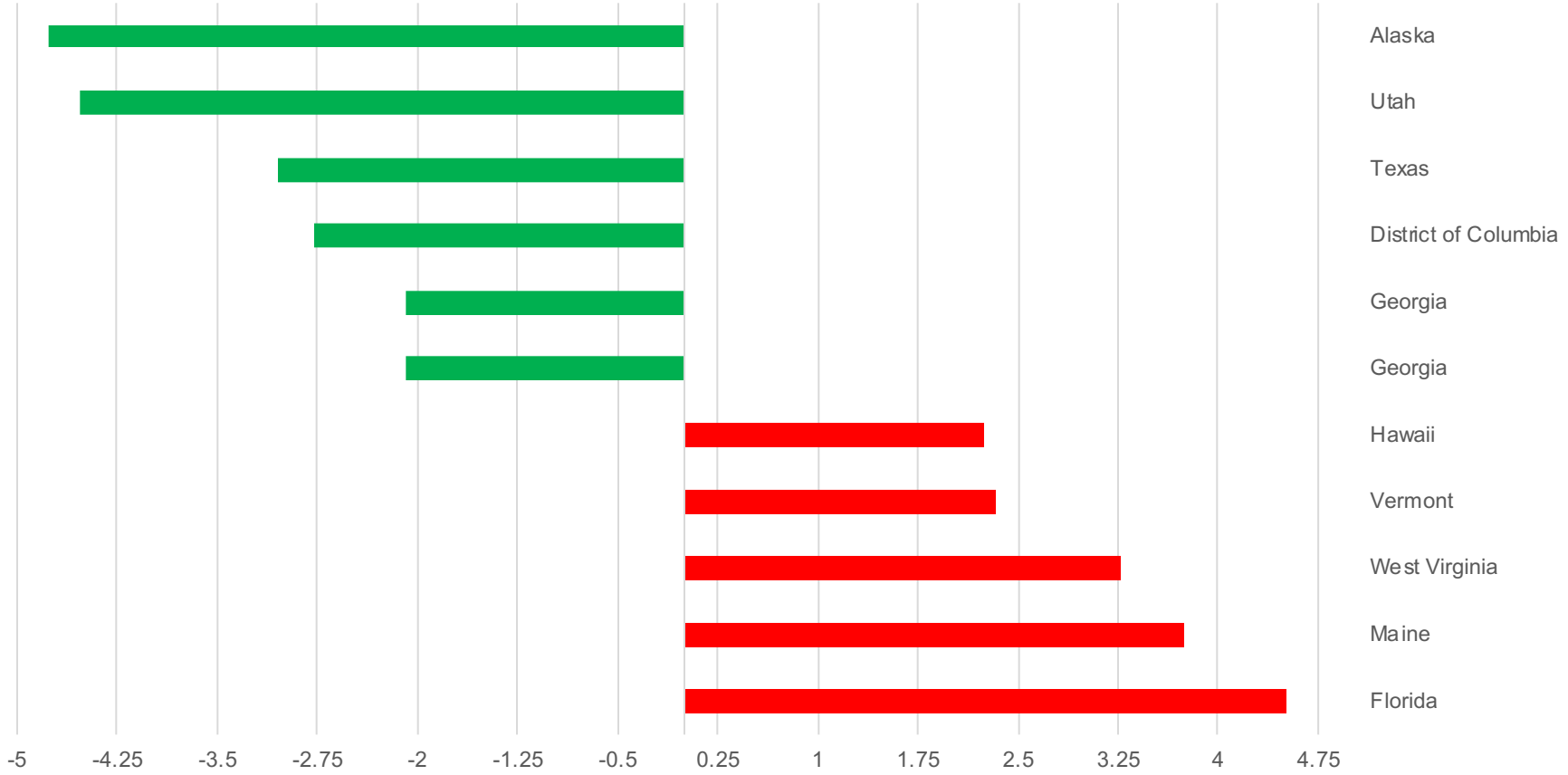
- **Data Sampling**

  Imbalance Dataset. In a fraud detection context data sets are typically very skew (e.g., 99 percent no fraudulent and below 1 percent fraudulent transactions).
- **Feature Selection**
- **Split the dataset into training and validation sets**
  - Cross validation to split the data into a test size of 20%
- **Imputed Missing Values**
  - 0 for numerical data
  - unknown for categorical data
- **Rescaled Data**
  - StandardScaler standardizes features by removing the mean and scaling to unit variance
- **Dummify Categorical Data**

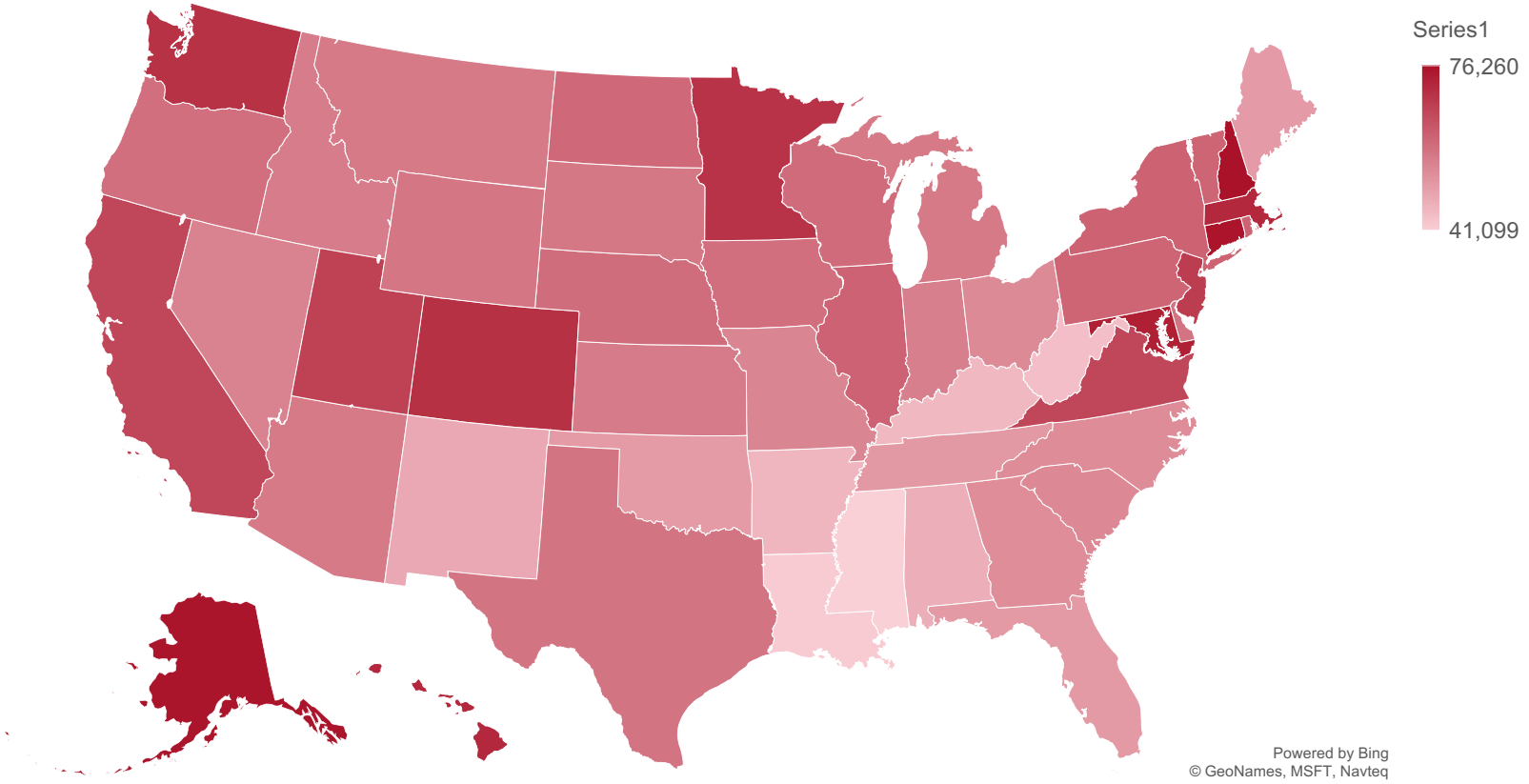
Population by State (millions)



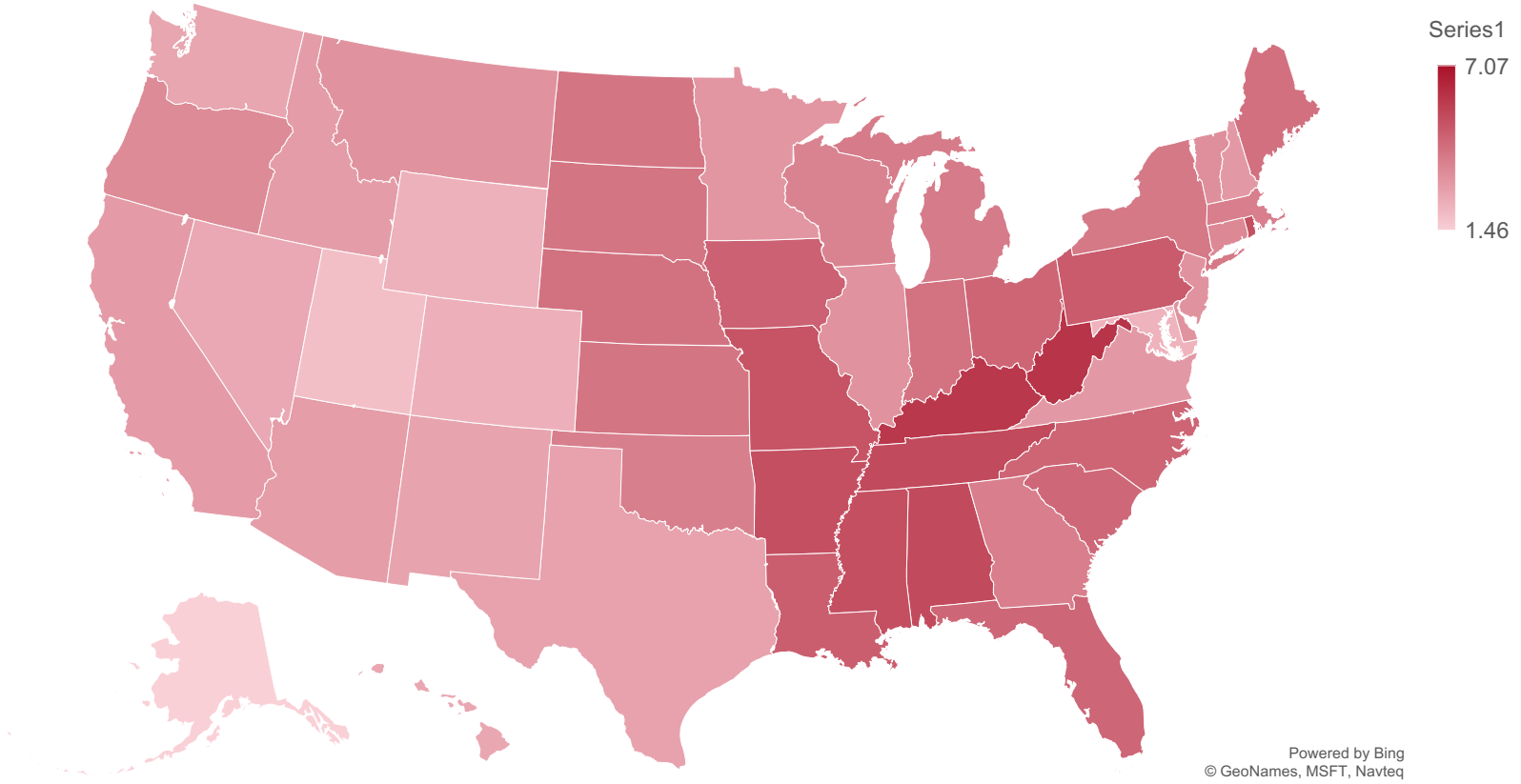
America’s Seniors and Youth



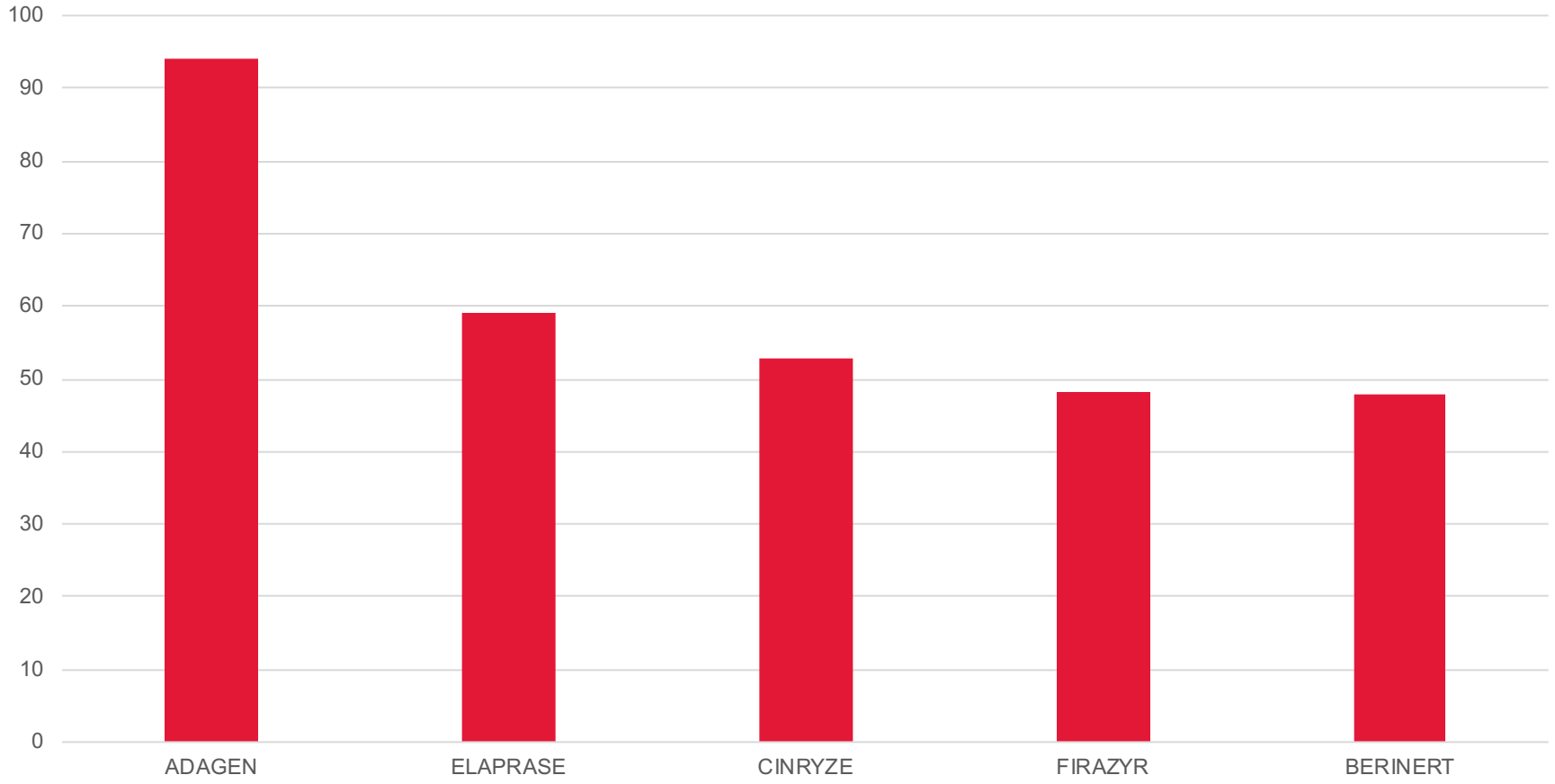
Median Household Income



Drug Claims per Capita



Cost per Claim, by Drug (000s)



Median Cost per Claim

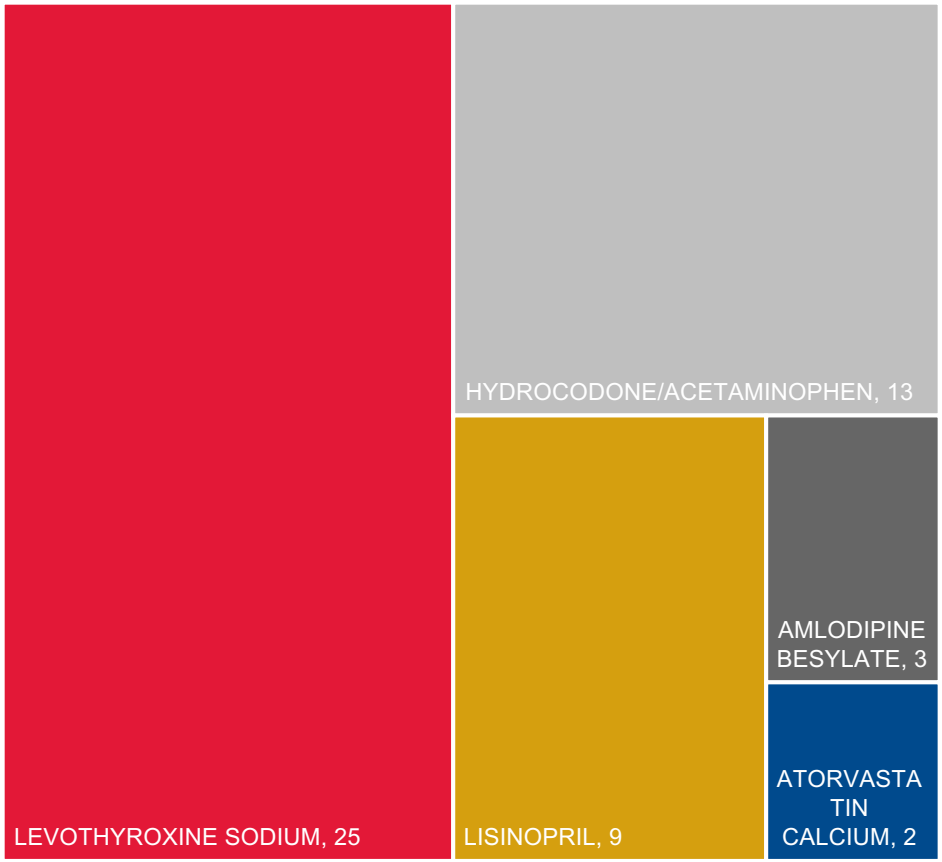
Over

- District of Columbia
- Delaware
- New Jersey
- New York
- Connecticut

Under

- Puerto Rico
- Iowa
- Arkansas
- Mississipi
- Rhode Island

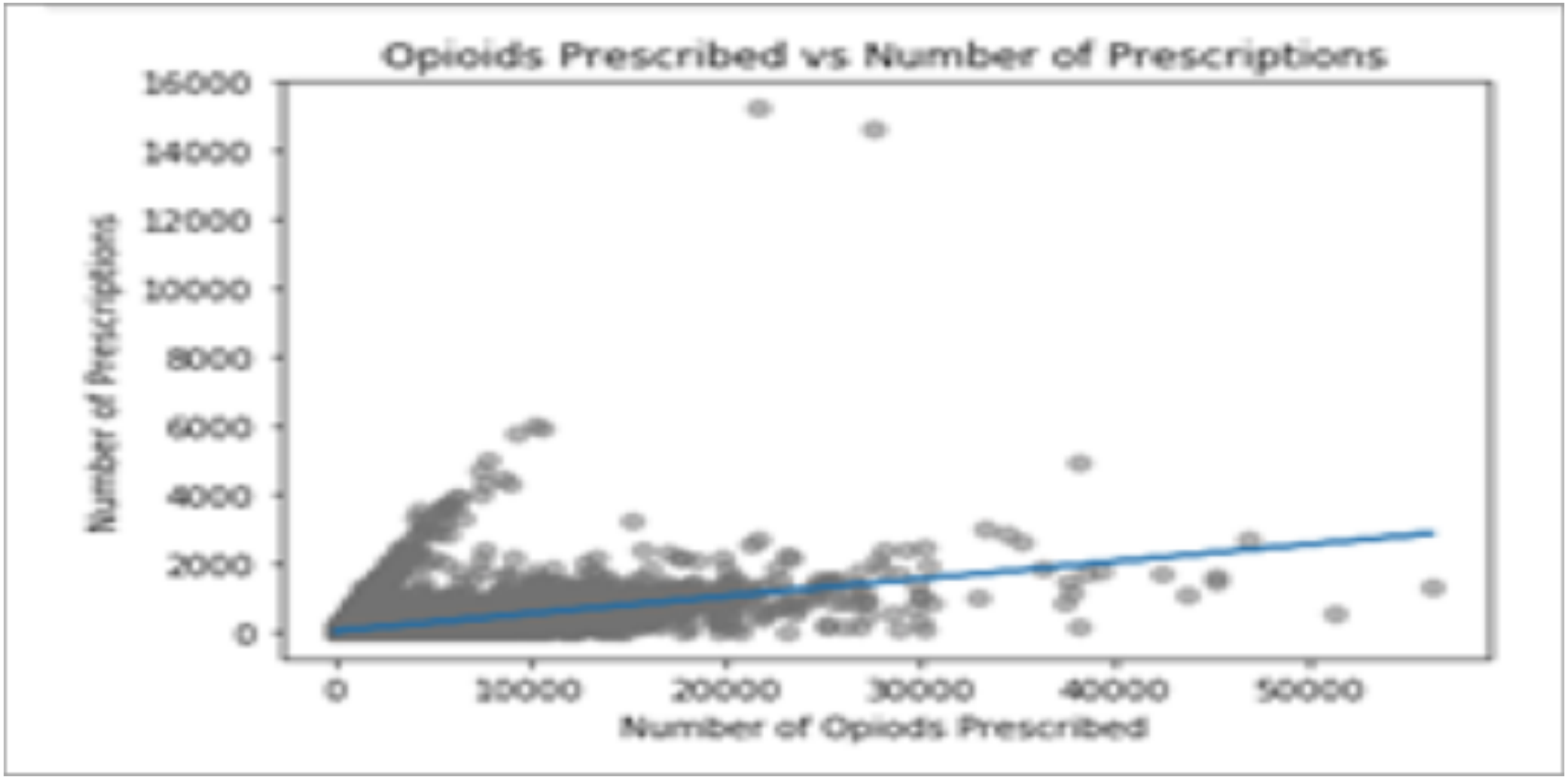
Top Drugs, By State



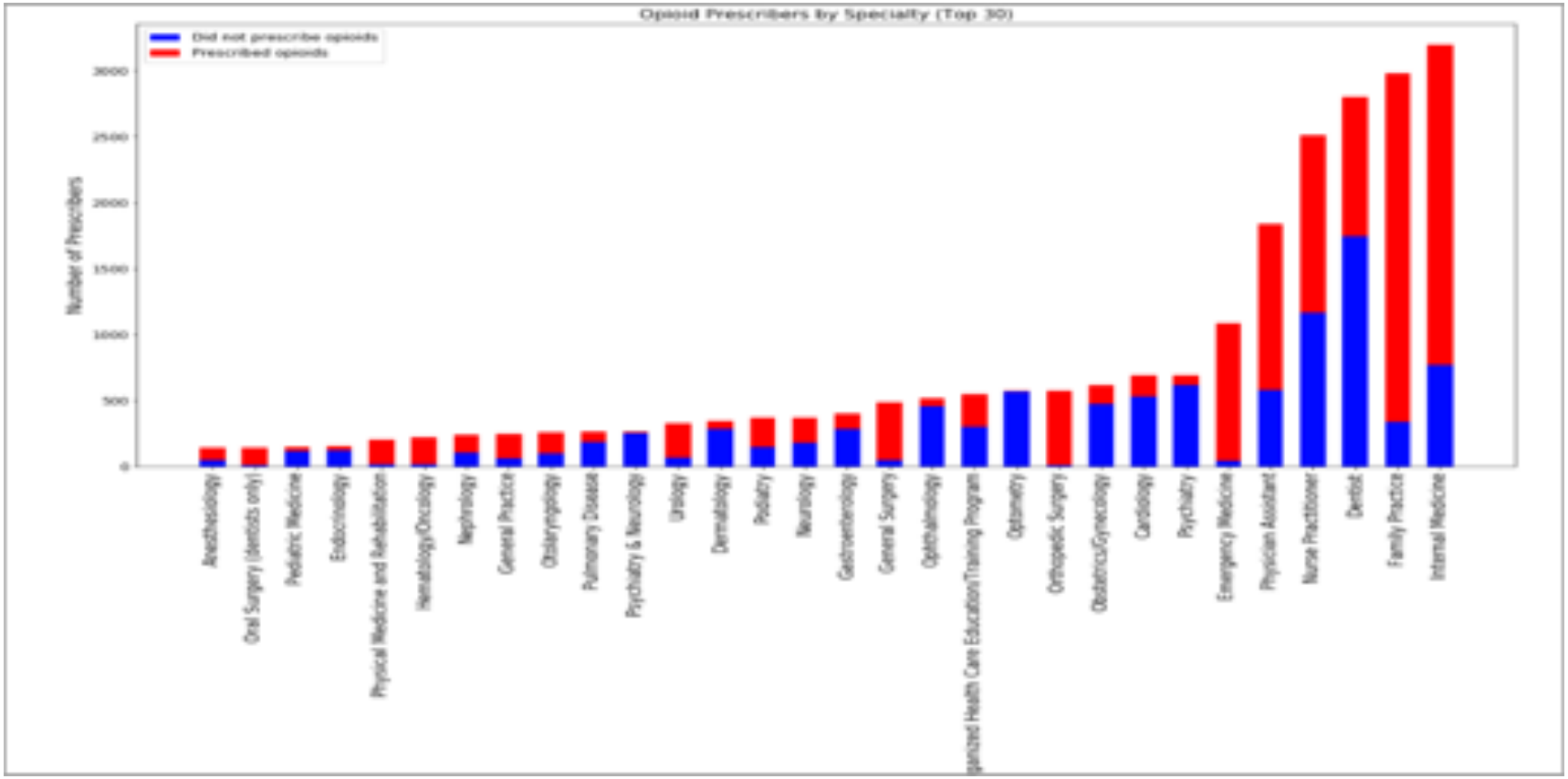
Opioid

- Texas
- Michigan
- Tennessee
- Alabama
- Indiana
- Kentucky
- Louisiana
- South Carolina
- Oklahoma
- Missouri
- Arkansas
- West Virginia
- Nevada

Opioid Drug Prescriptions Vs. Total Prescriptions



Opioid Prescribers by Specialty



Feature Engineering

- Part D data features
- Payment Data features
- Exclusive Data – Fraud Data Set
- Drug Data Features
- Geographic Social and Crime Data Features

- Feature types: Numeric and Categorical features

Key Features

- Payment Features

	first_name	last_name	city	state	Total_Payment_Sum
400215	JAMES	GAMMIE	Baltimore	MD	26843529.80
942922	STEPHEN	BURKHART	SAN ANTONIO	TX	22969245.26
153562	CHARLES	GOODIS	ALBUQUERQUE	NM	22880446.67
550533	KEVIN	FOLEY	Memphis	TN	17038512.75
387720	IVAN	OSORIO	KANSAS CITY	KS	8740393.58

Key Features

- Part D Features

```
npi                        893160
total_drug_cost_sum        893160
total_drug_cost_mean       893160
total_drug_cost_max        893160
total_claim_count_sum      893160
total_claim_count_mean     893160
total_claim_count_max      893160
total_day_supply_sum       893160
total_day_supply_mean      893160
total_day_supply_max       893160
dtype: int64
```

	npi	total_drug_cost_sum	total_drug_cost_mean	total_drug_cost_max	total_claim_count_sum	total_claim_count_mean	total_claim_
0	1003000126	20655.60	1377.040000	12110.20	310.0	20.666667	34.0
1	1003000142	120865.38	5493.880909	49748.43	1633.0	74.227273	358.0
2	1003000167	134.76	67.380000	71.33	32.0	16.000000	21.0
3	1003000282	328.04	164.020000	198.87	26.0	13.000000	14.0
4	1003000407	86421.07	1600.390185	15281.17	1741.0	32.240741	155.0

Key Features

- Merge Payment features with Part D features on last_name, first_name, city, state

```
In [62]: pay_partD_fpd = pd.merge (partD_allpd,payment_fpd2, how ='left', on = ['last_name','first_name','city','state'])

In [63]: pay_partD_fpd.head()
```

Out[63]:

	npi	total_drug_cost_sum	total_drug_cost_mean	total_drug_cost_max	total_claim_count_sum	total_claim_count_mean	total_claim_
0	1003000126	20655.60	1377.040000	12110.20	310.0	20.666667	34.0
1	1003000142	120865.38	5493.880909	49748.43	1633.0	74.227273	358.0
2	1003000167	134.76	67.380000	71.33	32.0	16.000000	21.0
3	1003000282	328.04	164.020000	198.87	26.0	13.000000	14.0
4	1003000407	86421.07	1600.390185	15281.17	1741.0	32.240741	155.0

```
In [64]: pay_partD_fpd.count()

Out[64]: npi                    893160
         total_drug_cost_sum    893160
         total_drug_cost_mean   893160
         total_drug_cost_max    893160
         total_claim_count_sum  893160
         total_claim_count_mean 893160
         total_claim_count_max  893160
         total_day_supply_sum   893160
         total_day_supply_mean  893160
         total_day_supply_max   893160
         city                   893160
         state                  893160
         last_name              893138
         first_name             893140
         Total_Payment_Sum      382080
         dtype: int64
```

Key Features

- Exclusive Data and Geo Crime Data

	npi	is_fraud
2	1922348218	1
21	1942476080	1
25	1275600959	1
28	1891731758	1
56	1902198435	1

```
print(npi_fraud_pd.dtypes)
```

```
npi          int64
is_fraud     int64
dtype: object
```

	state	Violent crime	Murder	Rape	Robbery	Aggravated assault	Property crime	Burglary	Larceny	Motor vehicle theft	Arson	AverageRank
0	AL	6.0	6.0	26.0	12.0	6.0	3.0	3.0	6.0	9.0	41.0	8
1	AK	5.0	22.0	1.0	14.0	5.0	5.0	14.0	7.0	1.0	20.0	5
2	AZ	37.0	38.0	30.0	36.0	39.0	30.0	31.0	27.0	38.0	35.0	40
3	AR	2.0	5.0	5.0	13.0	1.0	1.0	1.0	1.0	8.0	10.0	2
4	CA	29.0	28.0	33.0	21.0	34.0	32.0	29.0	37.0	16.0	15.0	31
5	CO	22.0	30.0	9.0	33.0	21.0	25.0	24.0	26.0	11.0	22.0	23
6	CT	18.0	20.0	36.0	15.0	27.0	35.0	35.0	36.0	20.0	17.0	27
7	DC	11.0	12.0	17.0	8.0	10.0	13.0	42.0	4.0	26.0	41.0	15
8	FL	26.0	25.0	32.0	30.0	24.0	23.0	23.0	21.0	29.0	30.0	29
9	GA	15.0	15.0	31.0	19.0	18.0	19.0	19.0	15.0	23.0	26.0	18
10	HI	40.0	40.0	37.0	38.0	42.0	33.0	39.0	31.0	28.0	5.0	39
11	IL	9.0	7.0	25.0	7.0	9.0	31.0	28.0	34.0	27.0	21.0	17

	state	AverageRank
0	AL	8
1	AK	5
2	AZ	40
3	AR	2
4	CA	31
5	CO	23
6	CT	27
7	DC	15
8	FL	29
9	GA	18
10	HI	39
11	IL	17

Key Features

- Opioids Data Features

	npi	Gender	State	Credentials	Specialty	ABILIFY	ACETAMINOPHEN.CODEINE	ACYCLOVIR	ADVAIR.DISKUS	AG
0	1710982582	M	TX	DDS	Dentist	0	0	0	0	0
1	1245278100	F	AL	MD	General Surgery	0	0	0	0	0
2	1427182161	F	NY	M.D.	General Practice	0	0	0	0	0
3	1669567541	M	AZ	MD	Internal Medicine	0	43	0	0	0
4	1679650949	M	NV	M.D.	Hematology/Oncology	0	0	0	0	0
5	1548580897	M	PA	DO	General Surgery	0	0	0	0	0
6	1437192002	M	NH	MD	Family Practice	0	0	0	25	0
7	1407113988	F	PA	RN, MSN, ANP-BC	Nurse Practitioner	0	0	0	0	0
8	1023260569	M	TX	O.D.	Optometry	0	0	0	0	0
9	1821106832	F	WI	MD	Internal Medicine	0	0	0	11	0
10	1609931914	F	PR	M.D.	General Practice	0	0	0	0	0
11	1659334472	M	TX	MD	General Surgery	0	0	0	0	0
12	1144205303	M	CO	MD	Family Practice	0	14	0	0	0
13	1548275050	M	OH	MD	Cardiology	0	0	0	0	0

Key Features

- Merge all Features together

```
print(FeaturesAll_pd.dtypes)
```

```
npi                          object
total_drug_cost_sum         float64
total_drug_cost_mean        float64
total_drug_cost_max         float64
total_claim_count_sum       float64
total_claim_count_mean      float64
total_claim_count_max       float64
total_day_supply_sum        float64
total_day_supply_mean       float64
total_day_supply_max        float64
city                         object
state                        object
last_name                    object
first_name                   object
Total_Payment_Sum           float64
is_fraud                    float64
AverageRank                 float64
Specialty                    object
is_Opioid_Prescriber        float64
claim_max-mean              float64
supply_max-mean             float64
drug_max-mean               float64
dtype: object
```

Key Features

- Explore the Drug_Weight and Speciality Weight

```
new_col_all
```

```
['LIDOCAINE_total_drug_cost',
 'LIDOCAINE_total_claim_count',
 'LIDOCAINE_total_day_supply',
 'FAMOTIDINE_total_claim_count',
 'FAMOTIDINE_total_day_supply',
 'NYSTOP_total_drug_cost',
 'NYSTOP_total_claim_count',
 'DULOXETINE HCL_total_drug_cost',
 'DULOXETINE HCL_total_claim_count',
 'DULOXETINE HCL_total_day_supply',
 'GABAPENTIN_total_drug_cost',
 'GABAPENTIN_total_claim_count',
 'OMEGA-3 ACID ETHYL ESTERS_total_drug_cost',
 'OMEGA-3 ACID ETHYL ESTERS_total_claim_count',
 'JANUMET_total_drug_cost',
 'JANUMET_total_claim_count',
 'JANUMET_total_day_supply',
 'CYCLOBENZAPRINE HCL_total_claim_count',
 'CYCLOBENZAPRINE HCL_total_day_supply',
 'METHADONE HCL_total_drug_cost',
 'FENTANYL_total_drug_cost',
 'FENTANYL_total_claim_count',
 'FENTANYL_total_day_supply',
 'SIMVASTATIN_total_drug_cost',
 'SIMVASTATIN_total_day_supply',
 'PROMETHAZINE HCL_total_drug_cost',
 'PROMETHAZINE HCL_total_claim_count',
 'PROMETHAZINE HCL_total_day_supply',
 'CELECOXIB_total_claim_count',
 'HYDROCODONE-ACETAMINOPHEN_total_drug_cost',
 'HYDROCODONE-ACETAMINOPHEN_total_claim_count',
 'HYDROCODONE-ACETAMINOPHEN_total_day_supply',
 'TIZANIDINE HCL_total_drug_cost',
```

```
In [131]: Create the Specialty Weight
          spec_dict =[]
          spec_fraud_1 = df_train[df_train['is_fraud']==1]['Specialty']

In [132]: from collections import Counter
          counts = Counter(spec_fraud_1)
          spec_dict =  dict(counts)

In [137]: FeaturesAll_pdl['Spec_Weight'] = FeaturesAll_pdl['Specialty'].map(lambda x: spec_dict.get(x, 0))

In [138]: df_train = FeaturesAll_pdl.ix[ix_train]
          df_valid = FeaturesAll_pdl.ix[ix_valid]
```

Fraud Labels 309 vs NPI Records 893162 without NaN
Fraud Labels 4798 vs All NPI Records 24964300 with NaN

```
Features_pd.count()
```

```
npi                          893162
total_drug_cost_sum          893162
total_drug_cost_mean         893162
total_drug_cost_max          893162
total_claim_count_sum        893162
total_claim_count_mean       893162
total_claim_count_max        893162
total_day_supply_sum         893162
total_day_supply_mean        893162
total_day_supply_max         893162
city                         893162
state                        893162
last_name                    893140
first_name                   893142
Total_Payment_Sum            382081
is_fraud                        309
AverageRank                  853873
dtype: int64
```

```
partD_pd.count()
```

```
npi                          24964300
nppes_provider_last_org_name 24963840
nppes_provider_first_name    24963985
nppes_provider_city          24964300
nppes_provider_state         24964300
specialty_description        24964300
description_flag             24964300
drug_name                    24964300
generic_name                 24964300
bene_count                    9776856
total_claim_count            24964300
total_30_day_fill_count      24964300
total_day_supply             24964300
total_drug_cost              24964300
bene_count_ge65               3450588
bene_count_ge65_suppress_flag 21513712
total_claim_count_ge65       14317150
ge65_suppress_flag           10647150
total_30_day_fill_count_ge65 14317150
total_day_supply_ge65        14317150
total_drug_cost_ge65         14317150
dtype: int64
```

```
npifraud_pd1.count()
```

```
NPI          4798
EXCLTYPE     4798
dtype: int64
```

Predictive Analytics Model for HealthCare Fraud Detection

- **Model Overview**

  In predictive analytics, the aim is to build an analytical model predicting a target measure of interest.

- **Models used in this projects**

  1. Logistic Regression
  2. Naïve Bayes
  3. Ensemble Methods: Random Forest / Extra Trees / Gradient Boosting

Predictive Analytics Model for HealthCare Fraud Detection

- **Models**

```
X= df_train[numerical_features].values
Y = df_train['is_fraud'].values
clf =  LogisticRegression(C=1e5, class_weight={0:1, 1:4000}, n_jobs=3)
clf.fit(X,Y)
y_p=clf.predict_proba(X)
```

```
/Users/my_macbook/anaconda3/lib/python3.6/site-packages/sklearn/linear_model/logistic.py:1228: UserWarning:

'n_jobs' > 1 does not have any effect when 'solver' is set to 'liblinear'. Got 'n_jobs' = 3.
```

```
params_0 = {'n_estimators': 100, 'max_depth': 8, 'min_samples_split': 3, 'learning_rate': 0.01}
params_1 = {'n_estimators': 500, 'max_depth': 10, 'min_samples_split': 5, 'class_weight' : {0:1, 1:4000}, 'n_jobs':5}

scaler = StandardScaler()

clfs = [
    LogisticRegression(C=1e5,class_weight={0:1, 1:4000}, n_jobs=3),

    GaussianNB(),

    ensemble.RandomForestClassifier(**params_1),

    ensemble.ExtraTreesClassifier(**params_1),

    ensemble.GradientBoostingClassifier(**params_0)

    ]
```

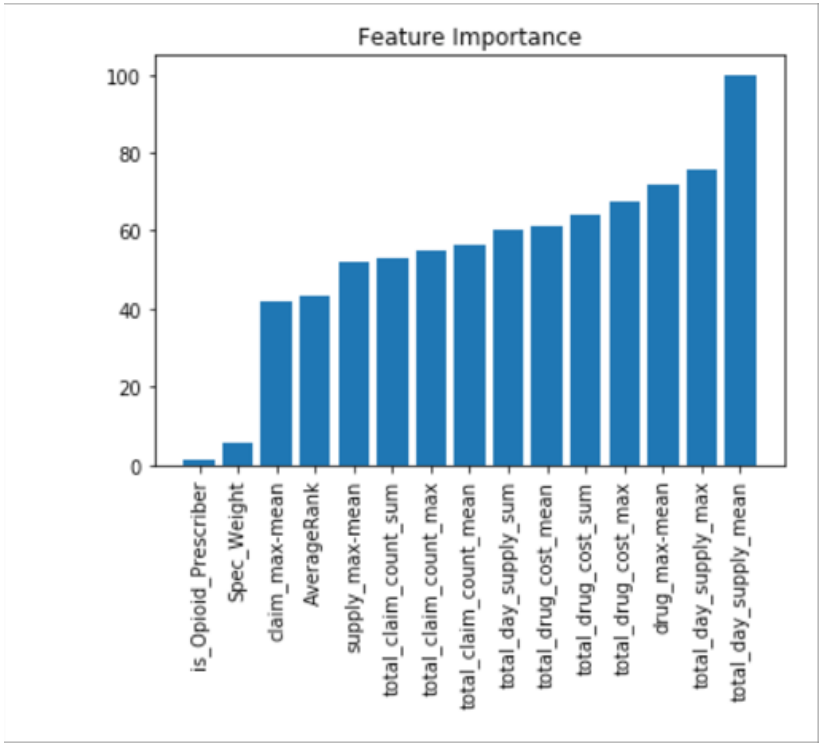
Evaluating Predictive Models

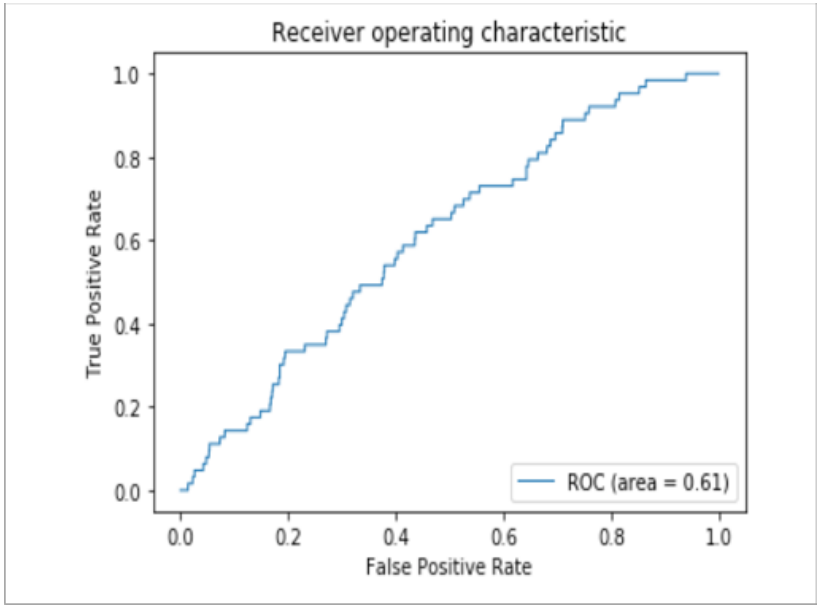
- **Splitting Up the Data Set and cross-validation**

- **Performance metrics and benchmarks**

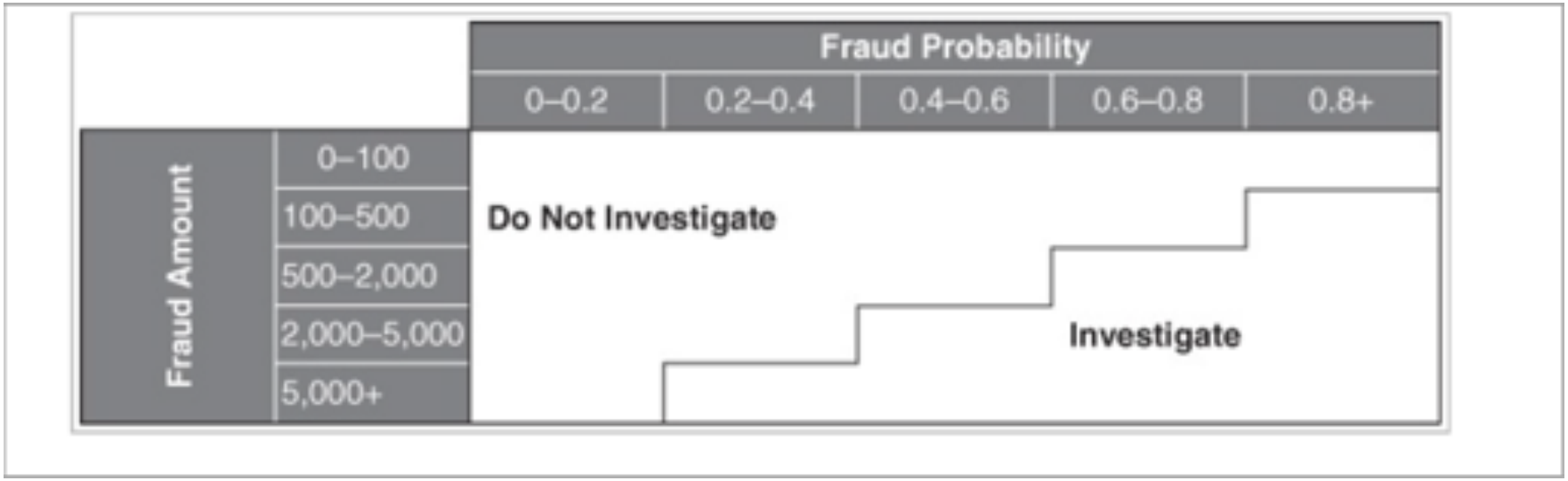
```
LogisticRegression:
Brier: 0.30044        Precision: 0.00044    Recall: 0.80952       F1:
 0.00088       auc: 0.56318   Accuracy: 0.35233

GaussianNB:
Brier: 0.00037        Precision: 0.00000    Recall: 0.00000       F1:
 0.00000       auc: 0.57728

RandomForestClassifier:
Brier: 0.08022        Precision: 0.00071    Recall: 0.11111       F1:
 0.00141       auc: 0.60865   Accuracy: 0.94431

ExtraTreesClassifier:
Brier: 0.23021        Precision: 0.00048    Recall: 0.65079       F1:
 0.00095       auc: 0.60590   Accuracy: 0.51934

GradientBoostingClassifier:
Brier: 0.00073        Precision: 0.00000    Recall: 0.00000       F1:
 0.00000       auc: 0.57620   Accuracy: 0.99926
```

Insights from the Model Performances



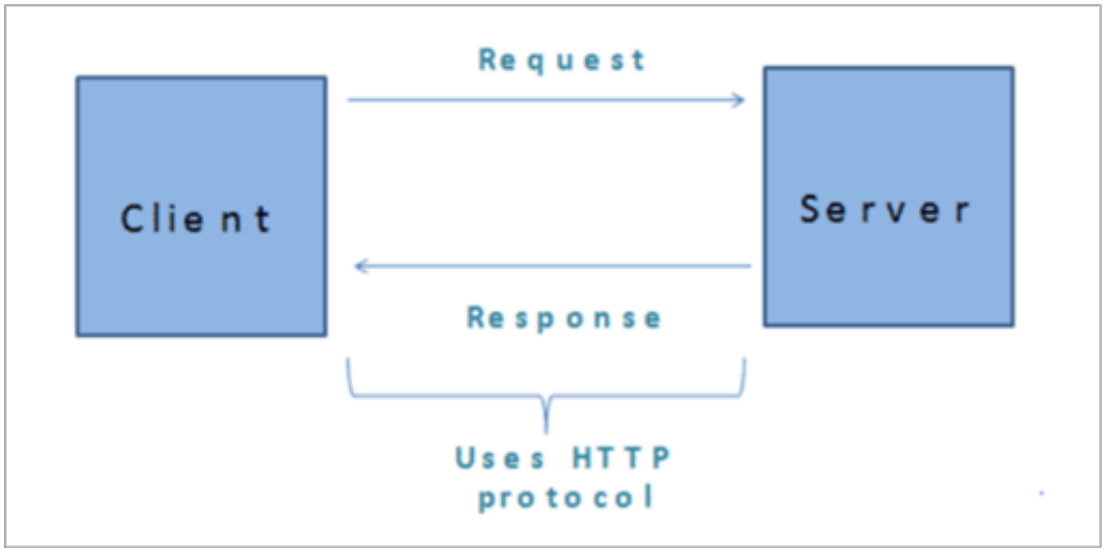


The Analytical Fraud Model Life Cycle



Product Deployment

1) Pilot a simple Flask-RESTful API to return a fraud indicator based on clients input using HTTP protocol



2) Offer customizable APIs based on client feedback to improve the product

Pilot RESTful API with Python and Flask

- Create Database
- Create Application Structure & Install Dependencies
- Create module
- Create GET endpoint to return the fraud flag based on NPI input
- Secure API with Auth0

How to market this product

- Partner with regulated bodies
- Targeted approach
- Pilot Partnership
- ROI

- Find additional product use cases to penetrate more markets and increase market awareness
  - Match individuals to doctors
  - Score doctor's and organizations for insurance underwriting
  - Create webinars, case studies, and articles about medical fraud