Importing libraries

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
import warnings
warnings.filterwarnings("ignore")
```

In [2]:
```
import os
os.getcwd()
```

Out[2]:
```
'C:\\Users\\TANYA\\Desktop\\Medical_Fraud_Data'
```

Reading the data

In [3]:
```
beneficiary_data = pd.read_csv("C:\\Users\\TANYA\\Desktop\\Medical_Fraud_Data\\Tr
inpatient_data = pd.read_csv("C:\\Users\\TANYA\\Desktop\\Medical_Fraud_Data\\Trai
outpatient_data = pd.read_csv("C:\\Users\\TANYA\\Desktop\\Medical_Fraud_Data\\Tra
train_data = pd.read_csv("C:\\Users\\TANYA\\Desktop\\Medical_Fraud_Data\\Train-15
```

Visualising the Data

In [4]:
```
beneficiary_data.head()
```

Out[4]:

	BeneID	DOB	DOD	Gender	Race	RenalDiseaseIndicator	State	County	NoOfMonths_Part
0	BENE11001	1943-01-01	NaN	1	1	0	39	230	
1	BENE11002	1936-09-01	NaN	2	1	0	39	280	
2	BENE11003	1936-08-01	NaN	1	1	0	52	590	
3	BENE11004	1922-07-01	NaN	1	1	0	39	270	
4	BENE11005	1935-09-01	NaN	1	1	0	24	680	

5 rows × 25 columns

In [5]:
```
inpatient_data.sample(5)
```

Out[5]:

	BeneID	ClaimID	ClaimStartDt	ClaimEndDt	Provider	InscClaimAmtReimbursed	At
4991	BENE28926	CLM67855	2009-09-13	2009-09-18	PRV57284	4000	
39835	BENE156914	CLM75191	2009-11-08	2009-11-16	PRV54339	7000	
21434	BENE89427	CLM79032	2009-12-09	2009-12-12	PRV57227	5000	
31937	BENE127924	CLM45485	2009-04-04	2009-04-09	PRV54676	5000	
19798	BENE83453	CLM56509	2009-06-21	2009-06-23	PRV52019	6000	

5 rows × 30 columns

In [6]:
```
outpatient_data.sample(5)
```

Out[6]:

	BeneID	ClaimID	ClaimStartDt	ClaimEndDt	Provider	InscClaimAmtReimbursed
183810	BENE63810	CLM271217	2009-03-28	2009-03-28	PRV52019	500
380786	BENE120082	CLM210744	2009-02-23	2009-02-23	PRV56511	30
22509	BENE17402	CLM300685	2009-04-13	2009-04-13	PRV53797	90
183699	BENE63770	CLM331397	2009-04-29	2009-04-30	PRV51574	400
321400	BENE103162	CLM522824	2009-08-13	2009-08-13	PRV54566	10

5 rows × 27 columns

In [7]:
```
train_data.sample(5)
```

Out[7]:

	Provider	PotentialFraud
4810	PRV57033	No
3619	PRV55538	No
2595	PRV54231	No
4195	PRV56251	No
999	PRV52248	No

Checking Type

In [8]:
```
beneficiary_data.dtypes
```

Out[8]:
```
BeneID                              object
DOB                                 object
DOD                                 object
Gender                               int64
Race                                 int64
RenalDiseaseIndicator               object
State                                int64
County                               int64
NoOfMonths_PartACov                  int64
NoOfMonths_PartBCov                  int64
ChronicCond_Alzheimer                int64
ChronicCond_Heartfailure             int64
ChronicCond_KidneyDisease            int64
ChronicCond_Cancer                   int64
ChronicCond_ObstrPulmonary           int64
ChronicCond_Depression               int64
ChronicCond_Diabetes                 int64
ChronicCond_IschemicHeart            int64
ChronicCond_Osteoporasis             int64
ChronicCond_rheumatoidarthritis      int64
ChronicCond_stroke                   int64
IPAnnualReimbursementAmt             int64
IPAnnualDeductibleAmt                int64
OPAnnualReimbursementAmt             int64
OPAnnualDeductibleAmt                int64
dtype: object
```

In [9]:
```
inpatient_data.dtypes
```

Out[9]:
```
BeneID                      object
ClaimID                     object
ClaimStartDt                object
ClaimEndDt                  object
Provider                    object
InscClaimAmtReimbursed       int64
AttendingPhysician          object
OperatingPhysician          object
OtherPhysician              object
AdmissionDt                 object
ClmAdmitDiagnosisCode       object
DeductibleAmtPaid          float64
DischargeDt                 object
DiagnosisGroupCode          object
ClmDiagnosisCode_1          object
ClmDiagnosisCode_2          object
ClmDiagnosisCode_3          object
ClmDiagnosisCode_4          object
ClmDiagnosisCode_5          object
ClmDiagnosisCode_6          object
ClmDiagnosisCode_7          object
ClmDiagnosisCode_8          object
ClmDiagnosisCode_9          object
ClmDiagnosisCode_10         object
ClmProcedureCode_1         float64
ClmProcedureCode_2         float64
ClmProcedureCode_3         float64
ClmProcedureCode_4         float64
ClmProcedureCode_5         float64
ClmProcedureCode_6         float64
dtype: object
```

In [10]:
```
outpatient_data.dtypes
```

Out[10]:
```
BeneID                        object
ClaimID                       object
ClaimStartDt                  object
ClaimEndDt                    object
Provider                      object
InscClaimAmtReimbursed         int64
AttendingPhysician            object
OperatingPhysician            object
OtherPhysician                object
ClmDiagnosisCode_1            object
ClmDiagnosisCode_2            object
ClmDiagnosisCode_3            object
ClmDiagnosisCode_4            object
ClmDiagnosisCode_5            object
ClmDiagnosisCode_6            object
ClmDiagnosisCode_7            object
ClmDiagnosisCode_8            object
ClmDiagnosisCode_9            object
ClmDiagnosisCode_10           object
ClmProcedureCode_1           float64
ClmProcedureCode_2           float64
ClmProcedureCode_3           float64
ClmProcedureCode_4           float64
ClmProcedureCode_5           float64
ClmProcedureCode_6           float64
DeductibleAmtPaid              int64
ClmAdmitDiagnosisCode         object
dtype: object
```

In [11]:
```
train_data.dtypes
```

Out[11]:
```
Provider            object
PotentialFraud      object
dtype: object
```

Number of rows and columns

In [12]:
```
print(inpatient_data.shape)
print(outpatient_data.shape)
print(train_data.shape)
print(beneficiary_data.shape)
```

```
(40474, 30)
(517737, 27)
(5410, 2)
(138556, 25)
```

Merging the Datasets

In [13]:
```
mergeddata1 = inpatient_data.append(outpatient_data,ignore_index=False)
```

In [14]:
```
mergeddata1.shape
```

Out[14]:
```
(558211, 30)
```

In [15]:
```
mergeddata2=pd.merge(mergeddata1,beneficiary_data, how='left', on=['BeneID'])
```

In [16]:
```
mergeddata2.shape
```

Out[16]:
```
(558211, 54)
```

In [17]:
```
finaltraindata=pd.merge(mergeddata2,train_data, how='inner',on=['Provider'])
```

In [18]:
```
finaltraindata.shape
```

Out[18]:
```
(558211, 55)
```

In [19]:
```
finaltraindata.describe()
```

Out[19]:

	InscClaimAmtReimbursed	DeductibleAmtPaid	ClmProcedureCode_1	ClmProcedureCode_2	C
count	558211.000000	557312.000000	23310.000000	5490.000000	
mean	997.012133	78.421085	5896.154612	4106.358106	
std	3821.534891	274.016812	3050.489933	2031.640878	
min	0.000000	0.000000	11.000000	42.000000	
25%	40.000000	0.000000	3848.000000	2724.000000	
50%	80.000000	0.000000	5363.000000	4019.000000	
75%	300.000000	0.000000	8669.000000	4439.000000	
max	125000.000000	1068.000000	9999.000000	9999.000000	

8 rows × 29 columns

In [20]:
```
finaltraindata.dtypes
```

Out[20]:
```
BeneID                              object
ClaimID                             object
ClaimStartDt                        object
ClaimEndDt                          object
Provider                            object
InscClaimAmtReimbursed               int64
AttendingPhysician                  object
OperatingPhysician                  object
OtherPhysician                      object
AdmissionDt                         object
ClmAdmitDiagnosisCode               object
DeductibleAmtPaid                  float64
DischargeDt                         object
DiagnosisGroupCode                  object
ClmDiagnosisCode_1                  object
ClmDiagnosisCode_2                  object
ClmDiagnosisCode_3                  object
ClmDiagnosisCode_4                  object
ClmDiagnosisCode_5                  object
ClmDiagnosisCode_6                  object
ClmDiagnosisCode_7                  object
ClmDiagnosisCode_8                  object
ClmDiagnosisCode_9                  object
ClmDiagnosisCode_10                 object
ClmProcedureCode_1                 float64
ClmProcedureCode_2                 float64
ClmProcedureCode_3                 float64
ClmProcedureCode_4                 float64
ClmProcedureCode_5                 float64
ClmProcedureCode_6                 float64
DOB                                 object
DOD                                 object
Gender                               int64
Race                                 int64
RenalDiseaseIndicator               object
State                                int64
County                               int64
NoOfMonths_PartACov                  int64
NoOfMonths_PartBCov                  int64
ChronicCond_Alzheimer                int64
ChronicCond_Heartfailure             int64
ChronicCond_KidneyDisease            int64
ChronicCond_Cancer                   int64
ChronicCond_ObstrPulmonary           int64
ChronicCond_Depression               int64
ChronicCond_Diabetes                 int64
ChronicCond_IschemicHeart            int64
ChronicCond_Osteoporasis             int64
ChronicCond_rheumatoidarthritis      int64
ChronicCond_stroke                   int64
IPAnnualReimbursementAmt             int64
IPAnnualDeductibleAmt                int64
OPAnnualReimbursementAmt             int64
OPAnnualDeductibleAmt                int64
```

```
PotentialFraud                         object
dtype: object
```

Feature Engineering

- Extracting No.Of ClaimDays

In [21]:
```
from datetime import datetime
date_format = "%d/%m/%Y"
from datetime import date
```

In [22]:
```
finaltraindata.ClaimStartDt = pd.to_datetime(finaltraindata.ClaimStartDt)
finaltraindata.ClaimEndDt = pd.to_datetime(finaltraindata.ClaimEndDt)
```

In [23]:
```
finaltraindata['No_of_claimdays']=finaltraindata['ClaimEndDt']-finaltraindata['Cl
```

In [24]:
```
finaltraindata['No_of_claimdays'].head(5)
```

Out[24]:
```
0     6 days
1    12 days
2    18 days
3     4 days
4     4 days
Name: No_of_claimdays, dtype: timedelta64[ns]
```

- Extracting Days in Hospital

In [25]:
```
finaltraindata.AdmissionDt = pd.to_datetime(finaltraindata.AdmissionDt)
finaltraindata.DischargeDt = pd.to_datetime(finaltraindata.DischargeDt)
```

In [26]:
```
finaltraindata['Days_in_Hospital']=finaltraindata['DischargeDt']-finaltraindata[
```

In [27]:
```
finaltraindata['Days_in_Hospital'].head(5)
```

Out[27]:
```
0     6 days
1    12 days
2    18 days
3     4 days
4     4 days
Name: Days_in_Hospital, dtype: timedelta64[ns]
```

In [28]:
```
finaltraindata['Days_in_Hospital'].equals(finaltraindata['No_of_claimdays'])
```

Out[28]:
```
False
```

In [29]:
```
finaltraindata['Days_in_Hospital'].sample(5)
```

Out[29]:
```
20790    NaT
300655   NaT
127847   NaT
356275   NaT
367241   NaT
Name: Days_in_Hospital, dtype: timedelta64[ns]
```

- Extracting Age

In [30]:
```
finaltraindata.DOB = pd.to_datetime(finaltraindata.DOB)
finaltraindata['DOB'].head(5)
```

Out[30]:
```
0   1943-01-01
1   1913-12-01
2   1922-10-01
3   1930-07-01
4   1925-09-01
Name: DOB, dtype: datetime64[ns]
```

In [31]:
```
finaltraindata['DOD'].count()
```

Out[31]:
```
4131
```

-- filling remaining values with a date to calculate age

In [32]:
```
finaltraindata['DOD']=finaltraindata['DOD'].fillna('2009-12-31')
```

In [33]:
```
finaltraindata['DOD'] = pd.to_datetime(finaltraindata.DOD)
```

In [34]:
```
finaltraindata['Age'] = finaltraindata['DOD']-finaltraindata['DOB']
finaltraindata['Age'] = ((finaltraindata['DOD'] - finaltraindata['DOB'])/365).dt.
```

-- Removing the Columns after extracting the features from them

In [35]:
```
finaltraindata=finaltraindata.drop(['ClaimStartDt','ClaimEndDt','AdmissionDt','Di
```

In [36]:
```
finaltraindata.dtypes
```

Out[36]:
```
BeneID                                  object
ClaimID                                 object
Provider                                object
InscClaimAmtReimbursed                   int64
AttendingPhysician                      object
OperatingPhysician                      object
OtherPhysician                          object
ClmAdmitDiagnosisCode                   object
DeductibleAmtPaid                      float64
DiagnosisGroupCode                      object
ClmDiagnosisCode_1                      object
ClmDiagnosisCode_2                      object
ClmDiagnosisCode_3                      object
ClmDiagnosisCode_4                      object
ClmDiagnosisCode_5                      object
ClmDiagnosisCode_6                      object
ClmDiagnosisCode_7                      object
ClmDiagnosisCode_8                      object
ClmDiagnosisCode_9                      object
ClmDiagnosisCode_10                     object
ClmProcedureCode_1                     float64
ClmProcedureCode_2                     float64
ClmProcedureCode_3                     float64
ClmProcedureCode_4                     float64
ClmProcedureCode_5                     float64
ClmProcedureCode_6                     float64
Gender                                   int64
Race                                     int64
RenalDiseaseIndicator                   object
State                                    int64
County                                   int64
NoOfMonths_PartACov                      int64
NoOfMonths_PartBCov                      int64
ChronicCond_Alzheimer                    int64
ChronicCond_Heartfailure                 int64
ChronicCond_KidneyDisease                int64
ChronicCond_Cancer                       int64
ChronicCond_ObstrPulmonary               int64
ChronicCond_Depression                   int64
ChronicCond_Diabetes                     int64
ChronicCond_IschemicHeart                int64
ChronicCond_Osteoporasis                 int64
ChronicCond_rheumatoidarthritis          int64
ChronicCond_stroke                       int64
IPAnnualReimbursementAmt                 int64
IPAnnualDeductibleAmt                    int64
OPAnnualReimbursementAmt                 int64
OPAnnualDeductibleAmt                    int64
PotentialFraud                          object
No_of_claimdays                857timedelta64[ns]
Days_in_Hospital                timedelta64[ns]
Age                                      int64
dtype: object
```

Type Conversion

In [37]:
```
finaltraindata['DeductibleAmtPaid']= finaltraindata['DeductibleAmtPaid'].fillna(f
```

In [38]:
```

finaltraindata.No_of_claimdays= finaltraindata.No_of_claimdays.astype('int64')
finaltraindata.DeductibleAmtPaid = finaltraindata.DeductibleAmtPaid.astype('int64
```

In [39]:
```
catcol = ('Race','Gender','RenalDiseaseIndicator','Provider','State','County','Ch
          'ChronicCond_Heartfailure','ChronicCond_KidneyDisease','ChronicCond_Car
          'ChronicCond_Depression','ChronicCond_Diabetes','ChronicCond_IschemicHe
          'ChronicCond_rheumatoidarthritis','ChronicCond_stroke','PotentialFraud'
```

In [40]:
```
for i in catcol:
    finaltraindata[i] = finaltraindata[i].astype('category')
```

In [41]:
```
finaltraindata.dtypes
```

Out[41]:
```
BeneID                                  object
ClaimID                                 object
Provider                              category
InscClaimAmtReimbursed                   int64
AttendingPhysician                      object
OperatingPhysician                      object
OtherPhysician                          object
ClmAdmitDiagnosisCode                   object
DeductibleAmtPaid                        int64
DiagnosisGroupCode                      object
ClmDiagnosisCode_1                      object
ClmDiagnosisCode_2                      object
ClmDiagnosisCode_3                      object
ClmDiagnosisCode_4                      object
ClmDiagnosisCode_5                      object
ClmDiagnosisCode_6                      object
ClmDiagnosisCode_7                      object
ClmDiagnosisCode_8                      object
ClmDiagnosisCode_9                      object
ClmDiagnosisCode_10                     object
ClmProcedureCode_1                     float64
ClmProcedureCode_2                     float64
ClmProcedureCode_3                     float64
ClmProcedureCode_4                     float64
ClmProcedureCode_5                     float64
ClmProcedureCode_6                     float64
Gender                                category
Race                                  category
RenalDiseaseIndicator                 category
State                                 category
County                                category
NoOfMonths_PartACov                      int64
NoOfMonths_PartBCov                      int64
ChronicCond_Alzheimer                 category
ChronicCond_Heartfailure              category
ChronicCond_KidneyDisease             category
ChronicCond_Cancer                    category
ChronicCond_ObstrPulmonary            category
ChronicCond_Depression                category
ChronicCond_Diabetes                  category
ChronicCond_IschemicHeart             category
ChronicCond_Osteoporasis              category
ChronicCond_rheumatoidarthritis       category
ChronicCond_stroke                    category
IPAnnualReimbursementAmt                 int64
IPAnnualDeductibleAmt                    int64
OPAnnualReimbursementAmt                 int64
OPAnnualDeductibleAmt                    int64
PotentialFraud                        category
No_of_claimdays                          int64
Days_in_Hospital               timedelta64[ns]
Age                                      int64
dtype: object
```

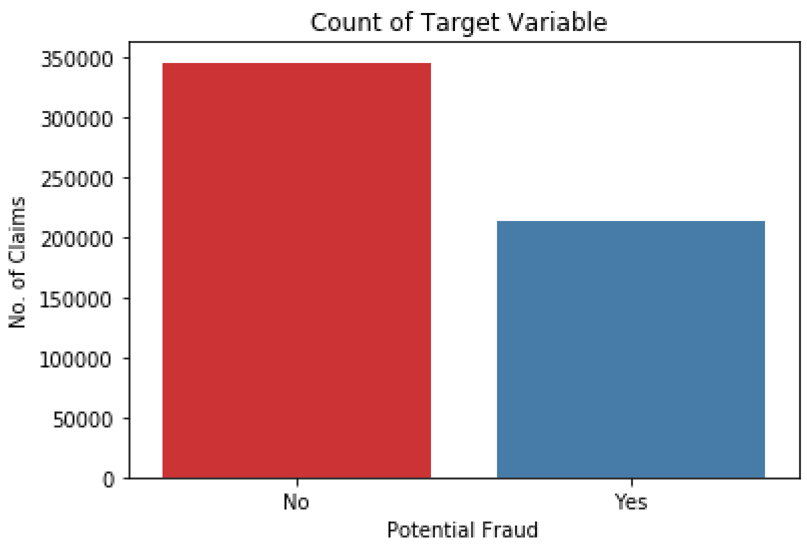
Data Analysis

In [42]:
```
finaltraindata['PotentialFraud'].value_counts()
```

Out[42]:
```
No     345415
Yes    212796
Name: PotentialFraud, dtype: int64
```

In [43]:
```
sns.countplot(x='PotentialFraud',data=finaltraindata,palette='Set1')
plt.title("Count of Target Variable ")
plt.xlabel('Potential Fraud')
plt.ylabel('No. of Claims')
```

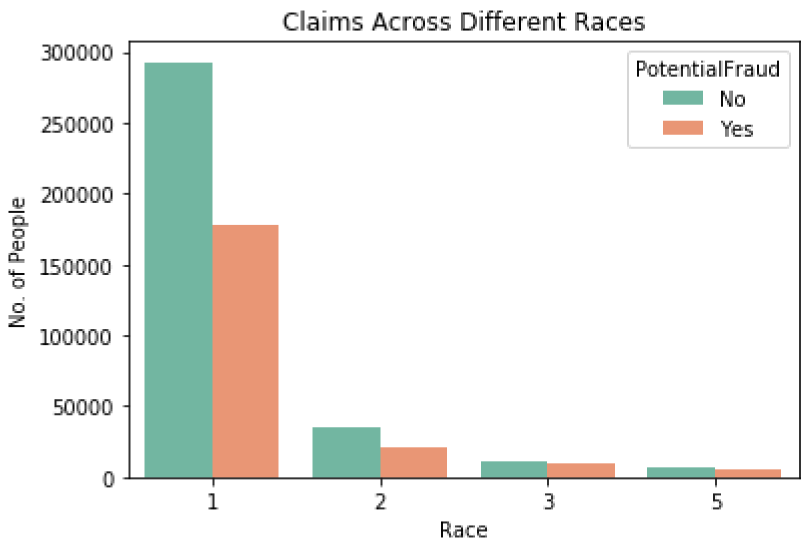
Out[43]:
```
Text(0, 0.5, 'No. of Claims')
```



We can see that total no. of FRAUD claims are significant and accounts upto 40% of total claims

In [44]:
```
sns.countplot(x="Race",hue="PotentialFraud",palette='Set2',data=finaltraindata)
plt.title("Claims Across Different Races")
plt.xlabel('Race')
plt.ylabel('No. of People')
```

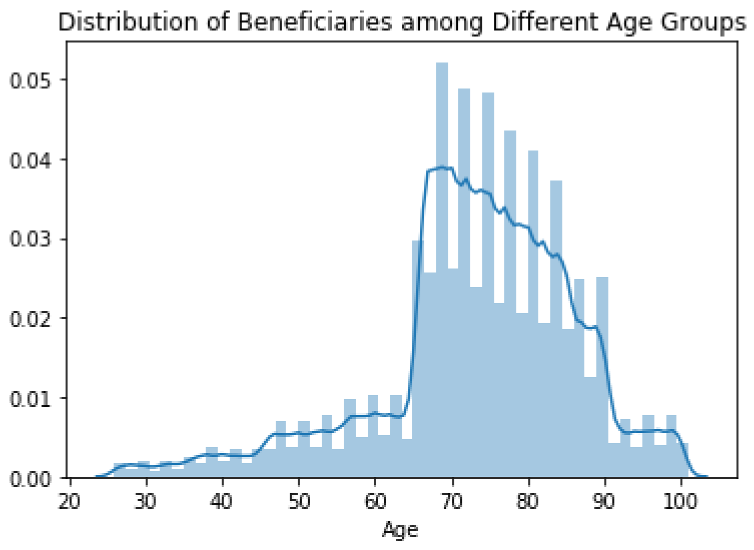
Out[44]:
```
Text(0, 0.5, 'No. of People')
```



Claims belonging to Race 3 having high probality of being Fraud

In [45]:
```
sns.distplot(finaltraindata['Age'])
plt.title('Distribution of Beneficiaries among Different Age Groups')
```

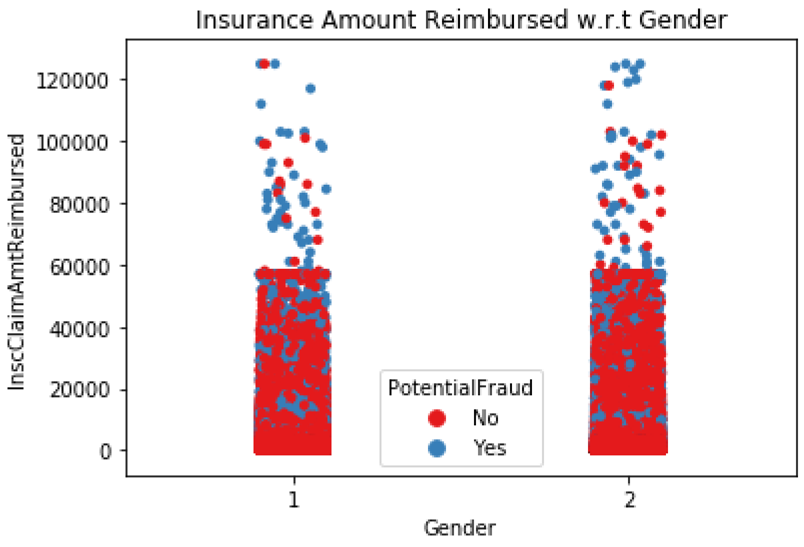
Out[45]:
```
Text(0.5, 1.0, 'Distribution of Beneficiaries among Different Age Groups')
```



Insurance Claims are mostly taken from the age group of 65-90

In [46]:
```
sns.stripplot(x="Gender", y="InscClaimAmtReimbursed", data=finaltraindata,jitter=
plt.title('Insurance Amount Reimbursed w.r.t Gender')
```

Out[46]:
```
Text(0.5, 1.0, 'Insurance Amount Reimbursed w.r.t Gender')
```

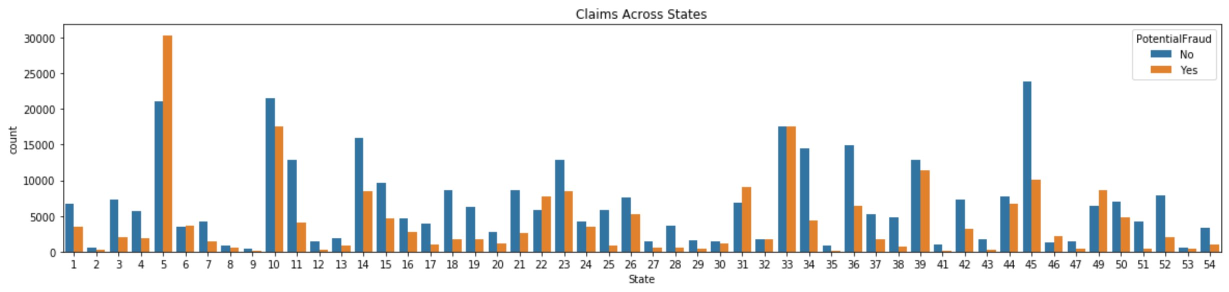


Insurance Fraud is seen in both the Gender

And if the Amount to be Reimbursed greater than 60000 for a claim, It has higher probability of Being a Fraud

In [47]:
```
plt.figure(figsize=(20,4))
sns.countplot(x='State',hue="PotentialFraud",data=finaltraindata)
plt.title('Claims Across States')
```

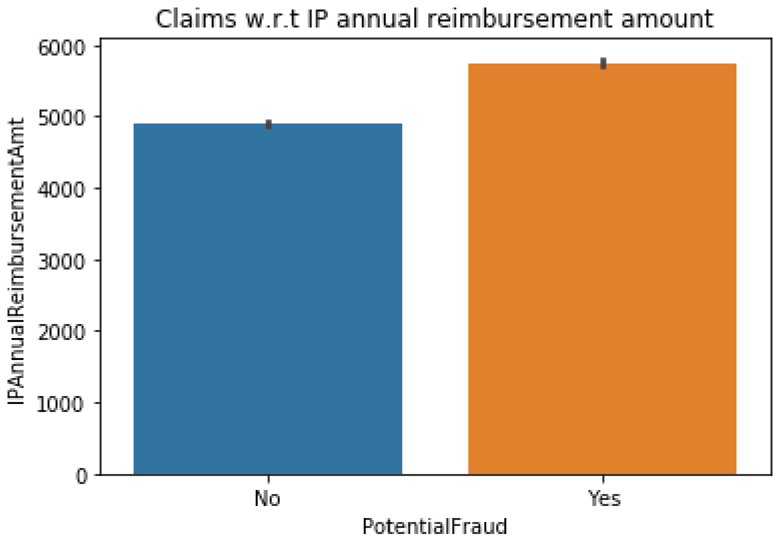
Out[47]:
```
Text(0.5, 1.0, 'Claims Across States')
```



Claims in State 5 & 33 are having higher probability of being Fraud

In [48]:
```
from numpy import median,mean
sns.barplot(x="PotentialFraud", y="IPAnnualReimbursementAmt", data=finaltraindata
plt.title('Claims w.r.t IP annual reimbursement amount')
```

Out[48]:
```
Text(0.5, 1.0, 'Claims w.r.t IP annual reimbursement amount')
```



If the Annual Reimbursement Amount for a provider is greater than 50000 the claims of the respective provider seems to be Fraud

In [ ]:
```
```