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

In [291…
```
df=pd.read_excel("insurance.xlsx")
```

In [293…
```
df                                  # to show the top 5 and last 5 rows
```

Out[293…

	months_as_customer	age	policy_number	policy_bind_date	policy_state	policy_csl	p
0	328	48	521585	2014-10-17	OH	250/500	
1	228	42	342868	2006-06-27	IN	250/500	
2	134	29	687698	2000-09-06	OH	100/300	
3	256	41	227811	1990-05-25	IL	250/500	
4	228	44	367455	2014-06-06	IL	500/1000	
...	...	...	...	...	...	...	
995	3	38	941851	1991-07-16	OH	500/1000	
996	285	41	186934	2014-01-05	IL	100/300	
997	130	34	918516	2003-02-17	OH	250/500	
998	458	62	533940	2011-11-18	IL	500/1000	
999	456	60	556080	1996-11-11	OH	250/500	

1000 rows × 39 columns

In [ ]:
```
# Title :-       "Insurance Claim Fraud Detection"
```

In [ ]:
```
# Domain:-      "Finance"
```

In [ ]:
```
# Discription:- "This project explores insurance claim data to find signs of fraud.
#               It uses simple analysis and charts to understand what makes a claim
```

In [ ]:
```
# Skills:-      "1 Python","2 Pandas","3 Statistics","4 Data manipulation","5 Data
#               "6 Data transformation","7 Data cleaning","8 Data visualization","9
```

Business Problem understanding

In [ ]:
```
#  To pridicting Insurance Claim Fraud Detection
```

Data understanding and exploration

In [15]:
```
df.shape              #  to check the total rows and columns
```

Out[15]:
```
(1000, 39)
```

In [ ]:
```
# Observe : there are 1000 rows and 39 columns
```

In [17]:
```
df.dtypes             # to check the data types of each columns
```

Out[17]:
```
months_as_customer                    int64
age                                   int64
policy_number                         int64
policy_bind_date             datetime64[ns]
policy_state                         object
policy_csl                           object
policy_deductable                     int64
policy_annual_premium               float64
umbrella_limit                        int64
insured_zip                           int64
insured_sex                          object
insured_education_level              object
insured_occupation                   object
insured_hobbies                      object
insured_relationship                 object
capital-gains                         int64
capital-loss                          int64
incident_date                datetime64[ns]
incident_type                        object
collision_type                       object
incident_severity                    object
authorities_contacted                object
incident_state                       object
incident_city                        object
incident_location                    object
incident_hour_of_the_day              int64
number_of_vehicles_involved           int64
property_damage                      object
bodily_injuries                       int64
witnesses                             int64
police_report_available              object
total_claim_amount                    int64
injury_claim                          int64
property_claim                        int64
vehicle_claim                         int64
auto_make                            object
auto_model                           object
auto_year                             int64
fraud_reported                       object
dtype: object
```

In [ ]:
```
# observe :  there are  17 int, 19 object,1 float,2 datatime datatypes columns
```

In [21]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 39 columns):
 #   Column                       Non-Null Count  Dtype         
---  ------                       --------------  -----         
 0   months_as_customer           1000 non-null   int64         
 1   age                          1000 non-null   int64         
 2   policy_number                1000 non-null   int64         
 3   policy_bind_date             1000 non-null   datetime64[ns]
 4   policy_state                 1000 non-null   object        
 5   policy_csl                   1000 non-null   object        
 6   policy_deductable            1000 non-null   int64         
 7   policy_annual_premium        1000 non-null   float64       
 8   umbrella_limit               1000 non-null   int64         
 9   insured_zip                  1000 non-null   int64         
 10  insured_sex                  1000 non-null   object        
 11  insured_education_level      1000 non-null   object        
 12  insured_occupation           1000 non-null   object        
 13  insured_hobbies              1000 non-null   object        
 14  insured_relationship         1000 non-null   object        
 15  capital-gains                1000 non-null   int64         
 16  capital-loss                 1000 non-null   int64         
 17  incident_date                1000 non-null   datetime64[ns]
 18  incident_type                1000 non-null   object        
 19  collision_type               1000 non-null   object        
 20  incident_severity            1000 non-null   object        
 21  authorities_contacted        909 non-null    object        
 22  incident_state               1000 non-null   object        
 23  incident_city                1000 non-null   object        
 24  incident_location            1000 non-null   object        
 25  incident_hour_of_the_day     1000 non-null   int64         
 26  number_of_vehicles_involved  1000 non-null   int64         
 27  property_damage              1000 non-null   object        
 28  bodily_injuries              1000 non-null   int64         
 29  witnesses                    1000 non-null   int64         
 30  police_report_available      1000 non-null   object        
 31  total_claim_amount           1000 non-null   int64         
 32  injury_claim                 1000 non-null   int64         
 33  property_claim               1000 non-null   int64         
 34  vehicle_claim                1000 non-null   int64         
 35  auto_make                    1000 non-null   object        
 36  auto_model                   1000 non-null   object        
 37  auto_year                    1000 non-null   int64         
 38  fraud_reported               1000 non-null   object        
dtypes: datetime64[ns](2), float64(1), int64(17), object(19)
memory usage: 304.8+ KB
```

In [ ]:
```
# Obesreve: to show the rows ,columns,null value and data type of each columns
```

In [23]:
```
df.isnull().sum()            # to check the null values of each columns
```

Out[23]:
```
months_as_customer             0
age                            0
policy_number                  0
policy_bind_date               0
policy_state                   0
policy_csl                     0
policy_deductable              0
policy_annual_premium          0
umbrella_limit                 0
insured_zip                    0
insured_sex                    0
insured_education_level        0
insured_occupation             0
insured_hobbies                0
insured_relationship           0
capital-gains                  0
capital-loss                   0
incident_date                  0
incident_type                  0
collision_type                 0
incident_severity              0
authorities_contacted         91
incident_state                 0
incident_city                  0
incident_location              0
incident_hour_of_the_day       0
number_of_vehicles_involved    0
property_damage                0
bodily_injuries                0
witnesses                      0
police_report_available        0
total_claim_amount             0
injury_claim                   0
property_claim                 0
vehicle_claim                  0
auto_make                      0
auto_model                     0
auto_year                      0
fraud_reported                 0
dtype: int64
```

In [ ]:
```
# observe : only one columns have null value "authorities_contacted column"
```

In [29]:
```
df.columns.tolist()
```

Out[29]:
```
['months_as_customer',
 'age',
 'policy_number',
 'policy_bind_date',
 'policy_state',
 'policy_csl',
 'policy_deductable',
 'policy_annual_premium',
 'umbrella_limit',
 'insured_zip',
 'insured_sex',
 'insured_education_level',
 'insured_occupation',
 'insured_hobbies',
 'insured_relationship',
 'capital-gains',
 'capital-loss',
 'incident_date',
 'incident_type',
 'collision_type',
 'incident_severity',
 'authorities_contacted',
 'incident_state',
 'incident_city',
 'incident_location',
 'incident_hour_of_the_day',
 'number_of_vehicles_involved',
 'property_damage',
 'bodily_injuries',
 'witnesses',
 'police_report_available',
 'total_claim_amount',
 'injury_claim',
 'property_claim',
 'vehicle_claim',
 'auto_make',
 'auto_model',
 'auto_year',
 'fraud_reported']
```

In [ ]:
```
# Obesreve: To show all columns name in a list
```

In [9]:
```
df.duplicated().sum()
```

Out[9]:
```
0
```

In [ ]:
```
# Obesreve: To check all rows for any duplicates values
```

In [31]:
```
df.head()
```

Out[31]:

	months_as_customer	age	policy_number	policy_bind_date	policy_state	policy_csl	poli
0	328	48	521585	2014-10-17	OH	250/500	
1	228	42	342868	2006-06-27	IN	250/500	
2	134	29	687698	2000-09-06	OH	100/300	
3	256	41	227811	1990-05-25	IL	250/500	
4	228	44	367455	2014-06-06	IL	500/1000	

5 rows × 39 columns

In [ ]:
```
# Obeserve: top 5 rows by default selects
```

In [33]:
```
df.tail()
```

Out[33]:

	months_as_customer	age	policy_number	policy_bind_date	policy_state	policy_csl	p
995	3	38	941851	1991-07-16	OH	500/1000	
996	285	41	186934	2014-01-05	IL	100/300	
997	130	34	918516	2003-02-17	OH	250/500	
998	458	62	533940	2011-11-18	IL	500/1000	
999	456	60	556080	1996-11-11	OH	250/500	

5 rows × 39 columns

In [ ]:
```
# Obeserve:  5 buttom rows by default selects
```

Explore each column wise

column 1= months_as_customer (that means Number of months the person has been a customer.)

In [35]:
```
df["months_as_customer"].dtypes
```

Out[35]:
```
dtype('int64')
```

In [ ]:
```
# Observe : this column is int Datatype,count/ discrete variable  and correct Datat
```

In [41]:
```
df["months_as_customer"].isnull().sum()
```

Out[41]:
```
0
```

In [ ]:
```
# Observe : in this column there is no null value
```

In [59]:
```
df["months_as_customer"].nunique()
```

Out[59]:
```
391
```

In [ ]:
```
# check the column has number of unique value
```

column 2 = "age"

In [43]:
```
df["age"].dtypes
```

Out[43]:
```
dtype('int64')
```

In [ ]:
```
# Observe : this column is int Datatype and correct Datatypes
```

In [45]:
```
df["age"].isnull().sum()
```

Out[45]:
```
0
```

In [ ]:
```
# Observe : in this column there is no null value
```

In [53]:
```
df["age"].nunique()
```

Out[53]:
```
46
```

In [ ]:
```
# Observe :the columns  have number of unique value
```

column 3= "policy_number" (that means Unique ID of the insurance policy)

In [51]:
```
df["policy_number"].dtypes
```

Out[51]:
```
dtype('int64')
```

In [ ]:
```
# Observe : this column is int Datatype,continuous variable and correct Datatypes
```

In [49]:
```
df["policy_number"].nunique()
```

Out[49]:
```
1000
```

In [ ]:
```
# each rows has unique value (if more then 30% of the rows values unique) there is
# simply drop them
```

column 4 ="policy_bind_date" (that means Date when the policy was created or started.)

In [67]:
```
df["policy_bind_date"].dtypes
```

Out[67]:
```
dtype('<M8[ns]')
```

In [ ]:
```
# Observe : this column is datetime Datatype and correct Datatypes
```

In [63]:
```
df["policy_bind_date"].nunique()
```

Out[63]:
```
951
```

In [ ]:
```
# almost each rows has unique value (if more then 30% of the rows values uni.) ther
# simply drop them
```

column 5 ="policy_state"

In [72]:
```
df["policy_state"].dtypes
```

Out[72]:
```
dtype('O')
```

In [ ]:
```
# Observe : this column is discrete variable,object  Datatype and correct Datatypes
```

In [74]:
```
df["policy_state"].value_counts()
```

Out[74]:
```
policy_state
OH    352
IL    338
IN    310
Name: count, dtype: int64
```

In [ ]:
```
# Observe : this column has three types of value OH contains more
```

In [76]:
```
df["policy_state"].isnull().sum()
```

Out[76]:
```
0
```

In [ ]:
```
# Observe : in this column there is no null value
```

In [78]:
```
df["policy_state"].nunique()
```

Out[78]:
```
3
```

In [ ]:
```
# Observe : this column has three types of value  contains
```

column 6= "policy_csl" that means Combined single limit for bodily injury/property damage

In [84]:
```
df["policy_csl"].dtypes
```

Out[84]:
```
dtype('O')
```

In [ ]:
```
# Observe : this column is discrete variable,object  Datatype and correct Datatypes
```

In [86]:
```
df["policy_csl"].value_counts()
```

Out[86]:
```
policy_csl
250/500     351
100/300     349
500/1000    300
Name: count, dtype: int64
```

In [ ]:
```
# Observe : this column has three types of value 250/500 contains more
```

In [88]:
```
df["policy_csl"].isnull().sum()
```

Out[88]:
```
0
```

In [ ]:
```
# Observe : in this column there is no null value
```

column 7="policy_deductable" (that means The deductible amount to be paid by the policyholder.)

In [92]:
```
df["policy_deductable"].dtypes
```

Out[92]:
```
dtype('int64')
```

In [ ]:
```
# Observe : this column is int Datatype,count/discrete variable and correct Datatyp
```

In [94]:
```
df["policy_deductable"].isnull().sum()
```

Out[94]:
```
0
```

In [ ]:
```
# Observe : in this column there is no null value
```

In [120…
```
df["policy_deductable"].unique()
```

Out[120…
```
array([1000, 2000,  500], dtype=int64)
```

In [ ]:
```
# Observe : this column has three types of value  contains
```

In [18]:
```
df["policy_deductable"].value_counts()
```

Out[18]:
```
policy_deductable
1000    351
500     342
2000    307
Name: count, dtype: int64
```

In [20]:
```
# Observe : this column has three types of value 1000 contains more
```

column 8= "policy_annual_premium" (Yearly premium paid by the insured person)

In [98]:
```
df["policy_annual_premium"].dtypes
```

Out[98]:
```
dtype('float64')
```

In [ ]:
```
# Observe : this column is float Datatype,continuous variable and correct Datatypes
```

In [100…
```
df["policy_annual_premium"].isnull().sum()
```

Out[100…
```
0
```

In [ ]:
```
# Observe : in this column there is no null value
```

In [126…
```
df["policy_annual_premium"].nunique()
```

Out[126…
```
991
```

In [ ]:
```
# observe : - almost each rows have unique value but this is impact the analysis  s
```

column 9= "umbrella_limit" (Extra liability coverage beyond regular policy limits)

In [108…
```
df["umbrella_limit"].dtypes
```

Out[108…
```
dtype('int64')
```

In [ ]:
```
# Observe : this column is int Datatype,continuous variable and correct Datatypes
```

In [110…
```
df["umbrella_limit"].isnull().sum()
```

Out[110…
```
0
```

In [ ]:
```
# Observe : in this column there is no null value
```

In [116…
```
df["umbrella_limit"].value_counts()
```

Out[116…
```
umbrella_limit
 0           798
 6000000      57
 5000000      46
 4000000      39
 7000000      29
 3000000      12
 8000000       8
 9000000       5
 2000000       3
 10000000      2
-1000000       1
Name: count, dtype: int64
```

In [ ]:
```
# observe : - there is one value in incorrect format
```

column 10= "insured_zip" (ZIP code of the insured person (very high cardinality))

In [130…
```
df["insured_zip"].dtypes
```

Out[130…
```
dtype('int64')
```

In [ ]:
```
# Observe : this column is int Datatype,continuous variable and correct Datatypes
```

In [136…
```
df["insured_zip"].unique()
```

Out[136…
```
995
```

In [ ]:
```
# Nearly unique; adds little value unless regional analysis so that simply drop the
```

In [224…
```
pd.set_option('display.max_columns', None)
#  observe : - show all rows and columns
```

columns 11="insured_sex" (Gender of the insured person (MALE/FEMALE))

In [276…
```
df["insured_sex"].dtypes
```

Out[276…
```
dtype('O')
```

In [ ]:
```
# Observe : this column is object  Datatype discrete variable  and correct Datatype
```

In [228…
```
df["insured_sex"].value_counts()
```

Out[228…
```
insured_sex
FEMALE    537
MALE      463
Name: count, dtype: int64
```

In [ ]:
```
# Observe : this column has FEMALE value  contains MORE
```

In [230…
```
df["insured_sex"].isnull().sum()
```

Out[230…
```
0
```

In [ ]:
```
# Observe : in this column there is no null value
```

In [232…
```
df["insured_sex"].unique()
```

Out[232…
```
array(['MALE', 'FEMALE'], dtype=object)
```

In [ ]:
```
# this columns contains 2 type
```

column 12 ="insured_education_level" (Highest education level of the insured)

In [234…
```
df["insured_education_level"].dtypes
```

Out[234…
```
dtype('O')
```

In [ ]:
```
# Observe : this column is object Datatype discrete variable  and correct Datatypes
```

In [236…
```
df["insured_education_level"].value_counts()
```

Out[236…
```
insured_education_level
JD             161
High School    160
Associate      145
MD             144
Masters        143
PhD            125
College        122
Name: count, dtype: int64
```

In [ ]:
```
# Observe : this column have JD  value  contains MORE
```

In [238…
```
df["insured_education_level"].isnull().sum()
```

Out[238…
```
0
```

In [ ]:
```
# Observe : in this column there is no null value
```

In [242…
```
df["insured_education_level"].nunique()
```

Out[242…
```
7
```

In [ ]:
```
# this columns contains 7 type
```

column 13="insured_occupation" (Job type of the insured)

In [248…
```
df["insured_occupation"].dtypes
```

Out[248…
```
dtype('O')
```

In [ ]:
```
# Observe : this column is object Datatype discrete variable  and correct Datatypes
```

In [246…
```
df["insured_occupation"].value_counts()
```

Out[246…
```
insured_occupation
machine-op-inspct    93
prof-specialty       85
tech-support         78
sales                76
exec-managerial      76
craft-repair         74
transport-moving     72
other-service        71
priv-house-serv      71
armed-forces         69
adm-clerical         65
protective-serv      63
handlers-cleaners    54
farming-fishing      53
Name: count, dtype: int64
```

In [ ]:
```
# Observe : this column have machine-op-inspct  value  contains MORE
```

In [244…
```
df["insured_occupation"].isnull().sum()
```

Out[244…
```
0
```

In [ ]:
```
# Observe : in this column there is no null value
```

In [250…
```
df["insured_occupation"].nunique()
```

Out[250…
```
14
```

In [ ]:
```
# this columns contains 14 type
```

column 14 = "insured_hobbies" (Listed hobbies of the insured person)

In [262…
```
df["insured_hobbies"].dtypes
```

Out[262…
```
dtype('O')
```

In [ ]:
```
# Observe : this column is object Datatype discrete variable  and correct Datatypes
```

In [264…
```
df["insured_hobbies"].value_counts()
```

Out[264…
```
insured_hobbies
reading           64
exercise          57
paintball         57
bungie-jumping    56
movies            55
golf              55
camping           55
kayaking          54
yachting          53
hiking            52
video-games       50
skydiving         49
base-jumping      49
board-games       48
polo              47
chess             46
dancing           43
sleeping          41
cross-fit         35
basketball        34
Name: count, dtype: int64
```

In [ ]:
```
# Observe : this column have reading value  contains MORE
```

In [254…
```
df["insured_hobbies"].isnull().sum()
```

Out[254…
```
0
```

In [ ]:
```
# Observe : in this column there is no null value
```

In [260…
```
df["insured_hobbies"].nunique()
```

Out[260…
```
20
```

In [ ]:
```
# this columns contains 20 type
```

column 15= "insured_relationship" (Relationship of the insured person in the household)

In [268…
```
df["insured_relationship"].dtypes
```

Out[268…
```
dtype('O')
```

In [ ]:
```
# Observe : this column is object Datatype discrete variable  and correct Datatypes
```

In [270…
```
df["insured_relationship"].value_counts()
```

Out[270…
```
insured_relationship
own-child         183
other-relative    177
not-in-family     174
husband           170
wife              155
unmarried         141
Name: count, dtype: int64
```

In [ ]:
```
# Observe : this column have own-child value  contains MORE
```

In [272…
```
df["insured_relationship"].isnull().sum()
```

Out[272…
```
0
```

In [ ]:
```
# Observe : in this column there is no null value
```

In [278…
```
df["insured_relationship"].nunique()
```

Out[278…
```
6
```

In [ ]:
```
# this columns contains 6 type
```

column 16= "capital-gains" (Income gained from capital assets)

In [280…
```
df["capital-gains"].dtypes
```

Out[280…
```
dtype('int64')
```

In [ ]:
```
# Observe : this column is int Datatype,continuous variable and correct Datatypes
```

In [282…
```
df["capital-gains"].isnull().sum()
```

Out[282… 0

In [ ]:
```
# Observe : in this column there is no null value
```

column 17="capital-loss"

In [284…
```
df["capital-loss"].dtypes
```

Out[284… dtype('int64')

In [ ]:
```
# Observe : this column is int Datatype,continuous variable and correct Datatypes
```

In [ ]:
```
df["capital-loss"].isnull().sum()
```

In [ ]:
```
# Observe : in this column there is no null value
```

column 18 = "incident_date"

In [286…
```
df["incident_date"].dtypes
```

Out[286… dtype('<M8[ns]')

In [ ]:
```
# Observe : this column is int Date time ,time series variable and correct Datatype
```

In [290…
```
df["incident_date"].isnull().sum()
```

Out[290… 0

In [ ]:
```
# Observe : in this column there is no null value
```

column 19="incident_type"

In [292…
```
df["incident_type"].dtypes
```

Out[292… dtype('O')

In [ ]:
```
# Observe : this column is object Datatype discrete variable  and correct Datatypes
```

In [294…
```
df["incident_type"].isnull().sum()
```

Out[294… 0

In [ ]:
```
# Observe : in this column there is no null value
```

In [296…
```
df["incident_type"].value_counts()
```

Out[296…
```
incident_type
Multi-vehicle Collision     419
Single Vehicle Collision    403
Vehicle Theft                94
Parked Car                   84
Name: count, dtype: int64
```

In [ ]:
```
# Observe : this column have Multi-vehicle Collision value  contains MORE
```

In [298…
```
df["incident_type"].nunique()
```

Out[298…
```
4
```

In [ ]:
```
# this columns contains 6 type
```

column 20="collision_type"

In [308…
```
df["collision_type"].dtypes
```

Out[308…
```
dtype('O')
```

In [ ]:
```
# Observe : this column is object Datatype discrete variable  and correct Datatypes
```

In [302…
```
df["collision_type"].isnull().sum()
```

Out[302…
```
0
```

In [ ]:
```
# Observe : in this column there is no null value
```

In [304…
```
df["collision_type"].value_counts()
```

Out[304…
```
collision_type
Rear Collision     292
Side Collision     276
Front Collision    254
?                  178
Name: count, dtype: int64
```

In [ ]:
```
# Observe : this column have rear Collision value  contains MORE
```

In [306…
```
df["collision_type"].nunique()
```

Out[306…
```
4
```

In [ ]:
```
# this columns contains 4 type
```

column 21 ="incident_severity"

In [310…
```
df["incident_severity"].dtypes
```

Out[310…
```
dtype('O')
```

In [ ]:
```
# Observe : this column is object Datatype discrete variable  and correct Datatypes
```

In [312…
```
df["incident_severity"].isnull().sum()
```

Out[312…
```
0
```

In [ ]:
```
# Observe : in this column there is no null value
```

In [318…
```
df["incident_severity"].value_counts()
```

Out[318…
```
incident_severity
Minor Damage      354
Total Loss        280
Major Damage      276
Trivial Damage     90
Name: count, dtype: int64
```

In [ ]:
```
# Observe : this column have Minor Damage  contains MORE value
```

In [316…
```
df["incident_severity"].nunique()
```

Out[316…
```
4
```

In [ ]:
```
# this columns contains 4 type
```

columns 22 = "authorities_contacted"

In [322…
```
df["authorities_contacted"].dtypes
```

Out[322…
```
dtype('O')
```

In [ ]:
```
# Observe : this column is object Datatype discrete variable  and correct Datatypes
```

In [320…
```
df["authorities_contacted"].isnull().sum()
```

Out[320…
```
91
```

In [ ]:
```
# Observe : in this column there is 91 null value
```

In [324…
```
df["authorities_contacted"].value_counts()
```

Out[324…
```
authorities_contacted
Police       292
Fire         223
Other        198
Ambulance    196
Name: count, dtype: int64
```

In [ ]:
```
# Observe : this column have  police value  contains MORE
```

In [328…
```
df["authorities_contacted"].unique()
```

Out[328…
```
array(['Police', nan, 'Fire', 'Other', 'Ambulance'], dtype=object)
```

In [ ]:
```
# this columns contains 4 type and some nan
```

column 23 ="incident_state"

In [330…
```
df["incident_state"].dtypes
```

Out[330…
```
dtype('O')
```

In [ ]:
```
# Observe : this column is object Datatype discrete variable  and correct Datatypes
```

In [332…
```
df["incident_state"].isnull().sum()
```

Out[332…
```
0
```

In [ ]:
```
# Observe : in this column there is no null value
```

In [336…
```
df["incident_state"].value_counts()
```

Out[336…
```
incident_state
NY    262
SC    248
WV    217
VA    110
NC    110
PA     30
OH     23
Name: count, dtype: int64
```

In [ ]:
```
# Observe : this column have NY value  contains MORE
```

In [340…
```
df["incident_state"].nunique()
```

Out[340…
```
7
```

In [ ]:
```
# this columns contains 7 type
```

column 24="incident_city"

In [346…
```
df["incident_city"].dtypes
```

Out[346…
```
dtype('O')
```

In [ ]:
```
# Observe : this column is object Datatype discrete variable  and correct Datatypes
```

In [348…
```
df["incident_city"].isnull().sum()
```

Out[348…
```
0
```

In [ ]:
```
# Observe : in this column there is no null value
```

In [342…
```
df["incident_city"].value_counts()
```

Out[342…
```
incident_city
Springfield    157
Arlington      152
Columbus       149
Northbend      145
Hillsdale      141
Riverwood      134
Northbrook     122
Name: count, dtype: int64
```

In [ ]:
```
# Observe : this column have Springfield  value  contains MORE
```

In [344…
```
df["incident_city"].nunique()
```

Out[344…
```
7
```

In [ ]:
```
# this columns contains 7 type
```

columns 25=" incident_location"

In [350…
```
df["incident_location"].dtypes
```

Out[350…
```
dtype('O')
```

In [ ]:
```
# Observe : this column is object Datatype discrete variable  and correct Datatypes
```

In [356…
```
df["incident_location"].nunique()
```

Out[356…
```
1000
```

In [ ]:
```
#each rows has unique value (if more then 30% of the rows values uni.) there is no
# simply drop them
```

column 26="incident_hour_of_the_day"

In [358…
```
df["incident_hour_of_the_day"].dtypes
```

Out[358…
```
dtype('int64')
```

In [ ]:
```
# Observe : this column is int  Datatype count/discrete variable  and correct Datat
```

In [360…
```
df["incident_hour_of_the_day"].value_counts()
```

Out[360…
```
incident_hour_of_the_day
17    54
3     53
0     52
23    51
16    49
13    46
10    46
4     46
6     44
9     43
14    43
21    42
18    41
12    40
19    40
7     40
15    39
22    38
8     36
20    34
5     33
2     31
11    30
1     29
Name: count, dtype: int64
```

In [ ]:
```
# Observe : this column have Springfield  value  contains MORE
```

In [362…
```
df["incident_hour_of_the_day"].isnull().sum()
```

Out[362…
```
0
```

In [ ]:
```
# Observe : in this column there is no null value
```

In [368…
```
df["incident_hour_of_the_day"].nunique()
```

Out[368…
```
24
```

In [ ]:
```
# this columns contains 24 type
```

columns 27="number_of_vehicles_involved"

In [370…
```
df["number_of_vehicles_involved"].dtypes
```

Out[370…
```
dtype('int64')
```

In [ ]:
```
# Observe : this column is int  Datatype count/discrete variable  and correct Datat
```

In [372…
```
df["number_of_vehicles_involved"].isnull().sum()
```

Out[372…
```
0
```

In [ ]:
```
# Observe : in this column there is no null value
```

In [374…
```
df["number_of_vehicles_involved"].value_counts()
```

Out[374…
```
number_of_vehicles_involved
1    581
3    358
4     31
2     30
Name: count, dtype: int64
```

In [ ]:
```
# Observe : this column have 1 value  contains MORE
```

In [376…
```
df["number_of_vehicles_involved"].nunique()
```

Out[376…
```
4
```

In [ ]:
```
# this columns contains 4 type
```

columns 28="property_damage"

In [378…
```
df["property_damage"].dtypes
```

Out[378…
```
dtype('O')
```

In [ ]:
```
# Observe : this column is object Datatype discrete variable  and correct Datatypes
```

In [380…
```
df["property_damage"].value_counts()
```

Out[380…
```
property_damage
?      360
NO     338
YES    302
Name: count, dtype: int64
```

In [ ]:
```
# Observe : this column have ? value  contains MORE
```

In [382…
```
df["property_damage"].isnull().sum()
```

Out[382…
```
0
```

In [ ]:
```
# Observe : in this column there is no null value
```

In [384…
```
df["property_damage"].unique()
```

Out[384…
```
array(['YES', '?', 'NO'], dtype=object)
```

In [ ]:
```
# this columns contains 2  type and one is ?
```

columns 29= "bodily_injuries"

In [386…
```
df["bodily_injuries"].dtypes
```

Out[386…
```
dtype('int64')
```

In [ ]:
```
# Observe : this column is int Datatype,count/ discrete variable  and correct Datat
```

In [388…
```
df["bodily_injuries"].isnull().sum()
```

Out[388…
```
0
```

In [ ]:
```
# Observe : in this column there is no null value
```

In [390…
```
df["bodily_injuries"].value_counts()
```

Out[390…
```
bodily_injuries
0    340
2    332
1    328
Name: count, dtype: int64
```

In [ ]:
```
# Observe : this column has 0 value  contains more
```

In [392…
```
df["bodily_injuries"].unique()
```

Out[392…
```
array([1, 0, 2], dtype=int64)
```

In [ ]:
```
# Observe : this column has 3 types of value  contains
```

column 30= "witnesses" (Number of witnesses present)

In [138…
```
df["witnesses"].dtypes
```

Out[138…
```
dtype('int64')
```

In [ ]:
```
# Observe : this column is int Datatype,count/ discrete variable  and correct Datat
```

In [140…
```
df["witnesses"].unique()
```

Out[140…
```
array([2, 0, 3, 1], dtype=int64)
```

In [ ]:
```
# Observe : this column has four types of value  contains
```

In [142…
```
df["witnesses"].isnull().sum()
```

Out[142…
```
0
```

In [ ]:
```
# Observe : in this column there is no null value
```

column 31= "police_report_available"

In [148…
```
df["police_report_available"].dtypes
```

Out[148…
```
dtype('O')
```

In [ ]:
```
# Observe : this column is discrete variable,object  Datatype and correct Datatypes
```

In [144…
```
df["police_report_available"].value_counts()
```

Out[144…
```
police_report_available
?      343
NO     343
YES    314
Name: count, dtype: int64
```

In [ ]:
```
# Observe : this column has three types of value  contains
```

In [146…
```
df["police_report_available"].isnull().sum()
```

Out[146…
```
0
```

In [ ]:
```
# Observe : in this column there is no null value
```

column 32= "total_claim_amount" (Total amount claimed from the insurance)

In [150…
```
df["total_claim_amount"].dtypes
```

Out[150… dtype('int64')

In [ ]:
```
# Observe : this column is int Datatype,continuous variable and correct Datatypes
```

In [152…
```
df["total_claim_amount"].isnull().sum()
```

Out[152… 0

In [ ]:
```
# Observe : in this column there is no null value
```

column 33 ="injury_claim" (Amount claimed for injuries)

In [158…
```
df["injury_claim"].dtypes
```

Out[158… dtype('int64')

In [ ]:
```
# Observe : this column is int Datatype,continuous variable and correct Datatypes
```

In [160…
```
df["injury_claim"].isnull().sum()
```

Out[160… 0

In [ ]:
```
# Observe : in this column there is no null value
```

In [164…
```
df["injury_claim"].nunique()
```

Out[164… 638

In [ ]:
```
# observe : - almost each rows have unique value but this is impact the analysis  s
```

column 34 ="property_claim" (Claim amount for property damage.)

In [ ]:
```
df["property_claim"].dtypes
```

In [ ]:
```
# Observe : this column is int Datatype,continuous variable and correct Datatypes
```

In [178…
```
df["property_claim"].isnull().sum()
```

Out[178… 0

In [ ]:
```
# Observe : in this column there is no null value
```

column 35= "vehicle_claim" (Claim amount for vehicle repair or replacement.)

In [170…
```
df["vehicle_claim"].dtypes
```

Out[170…
```
dtype('int64')
```

In [ ]:
```
# Observe : this column is int Datatype,continuous variable and correct Datatypes
```

In [174…
```
df["vehicle_claim"].isnull().sum()
```

Out[174…
```
0
```

In [ ]:
```
# Observe : in this column there is no null value
```

column 36 = "auto_make" (Car manufacturer)

In [182…
```
df["auto_make"].dtypes
```

Out[182…
```
dtype('O')
```

In [ ]:
```
# Observe : this column is object Datatype,discrete variable and correct Datatypes
```

In [184…
```
df["auto_make"].isnull().sum()
```

Out[184…
```
0
```

In [ ]:
```
# Observe : in this column there is no null value
```

In [190…
```
df["auto_make"].value_counts()
```

Out[190…
```
auto_make
Saab          80
Dodge         80
Suburu        80
Nissan        78
Chevrolet     76
Ford          72
BMW           72
Toyota        70
Audi          69
Accura        68
Volkswagen    68
Jeep          67
Mercedes      65
Honda         55
Name: count, dtype: int64
```

In [192…
```
# Observe : this column has 14 types of value  contains
```

column 37= "auto_model" (Model of the car.)

In [194…
```
df["auto_model"].dtypes
```

Out[194…
```
dtype('O')
```

In [ ]:
```
# Observe : this column is object Datatype,discrete variable and correct Datatypes
```

In [196…
```
df["auto_model"].isnull().sum()
```

Out[196…
```
0
```

In [ ]:
```
# Observe : in this column there is no null value
```

In [198…
```
df["auto_model"].value_counts()
```

Out[198…
```
auto_model
RAM               43
Wrangler          42
A3                37
Neon              37
MDX               36
Jetta             35
Passat            33
A5                32
Legacy            32
Pathfinder        31
Malibu            30
92x               28
Camry             28
Forrestor         28
F150              27
95                27
E400              27
93                25
Grand Cherokee    25
Escape            24
Tahoe             24
Maxima            24
Ultima            23
X5                23
Highlander        22
Civic             22
Silverado         22
Fusion            21
ML350             20
Impreza           20
Corolla           20
TL                20
CRV               20
C300              18
3 Series          18
X6                16
M5                15
Accord            13
RSX               12
Name: count, dtype: int64
```

In [ ]:
```
# Observe : this column has 14 types of value  contains
```

column 38= "auto_year" (Year of the car model)

In [200…
```
df["auto_year"].dtypes
```

Out[200…
```
dtype('int64')
```

In [ ]:
```
# Observe : this column is object Datatype,count/discrete variable and correct Data
```

In [202…
```
df["auto_year"].value_counts()
```

Out[202…
```
auto_year
1995    56
1999    55
2005    54
2006    53
2011    53
2007    52
2003    51
2009    50
2010    50
2013    49
2002    49
2015    47
1997    46
2012    46
2008    45
2014    44
2001    42
2000    42
1998    40
2004    39
1996    37
Name: count, dtype: int64
```

In [218…
```
df["auto_year"].isnull().sum()
```

Out[218…
```
0
```

In [ ]:
```
# Observe : in this column there is no null value
```

In [ ]:
```
# Observe : this column has  1995 model of value  contains more  or less in 1996
```

In [206…
```
df["auto_year"].nunique()
```

Out[206…
```
21
```

In [ ]:
```
# Observe : this column has 21 types of value  contains
```

column 39= "fraud_reported"

In [208…
```
df["fraud_reported"].dtypes
```

Out[208…
```
dtype('O')
```

In [220…
```
# Observe : this column is object Datatype,discrete variable and correct Datatypes
```

In [210…
```
df["fraud_reported"].isnull().sum()
```

Out[210…
```
0
```

In [ ]:
```
# Observe : in this column there is no null value
```

In [212…
```
df["fraud_reported"].value_counts()
```

Out[212…
```
fraud_reported
N    753
Y    247
Name: count, dtype: int64
```

In [ ]:
```
# Observe : this column has  N  value  contains more  or less in Y
```

In [214…
```
df["fraud_reported"].nunique()
```

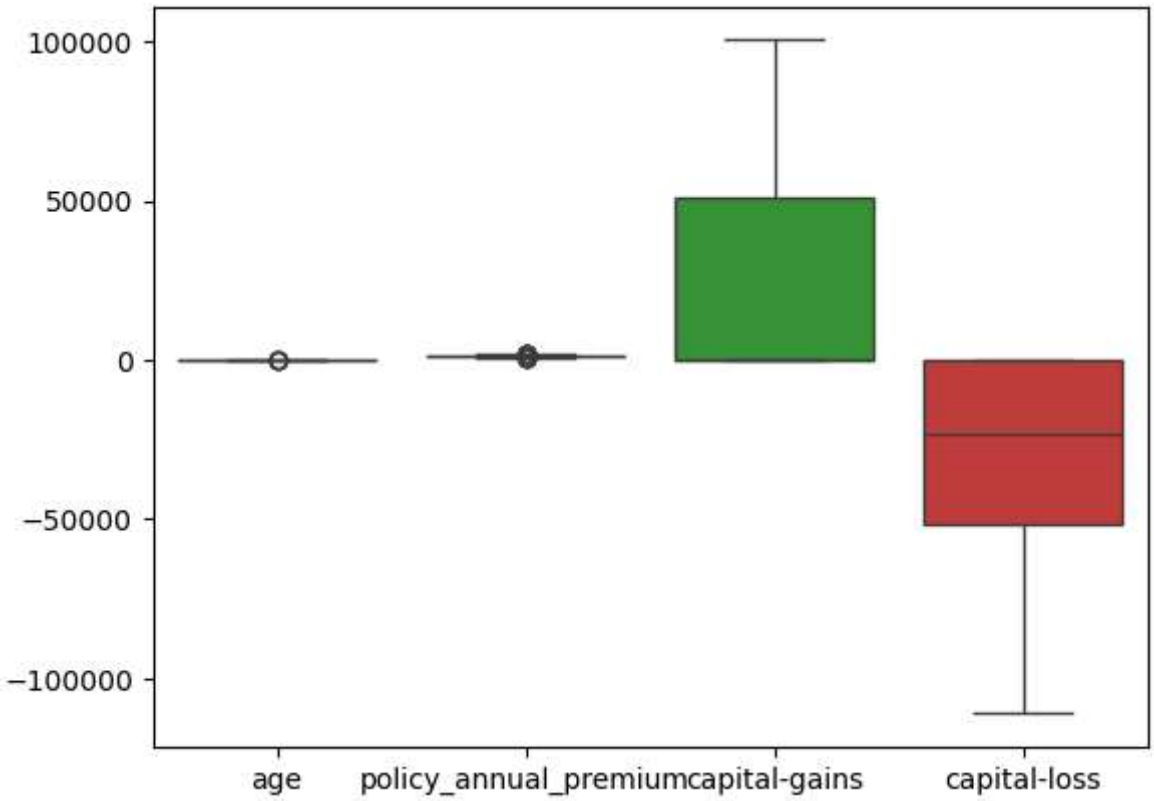
Out[214…
```
2
```

In [ ]:
```
# Observe : this column has 2 types of value  contains
```

In [22]:
```
# seperate the continuous ,timesries,discrete variable
continuous=["age","policy_annual_premium","capital-gains","capital-loss","total_cla
           "property_claim","vehicle_claim"]
timeseries=["incident_date"]
discrete=["months_as_customer","policy_state","policy_csl","policy_deductable","umb
        "insured_education_level","insured_occupation","insured_hobbies","insured_r
        ,"collision_type", "incident_severity","authorities_contacted","incident_st
        "incident_hour_of_the_day","number_of_vehicles_involved","property_damage",
        "bodily_injuries" ,"witnesses","police_report_available","auto_make","auto_
          "fraud_reported"]
```

In [34]:
```
sns.boxplot(df[["age","policy_annual_premium","capital-gains","capital-loss"]])
```

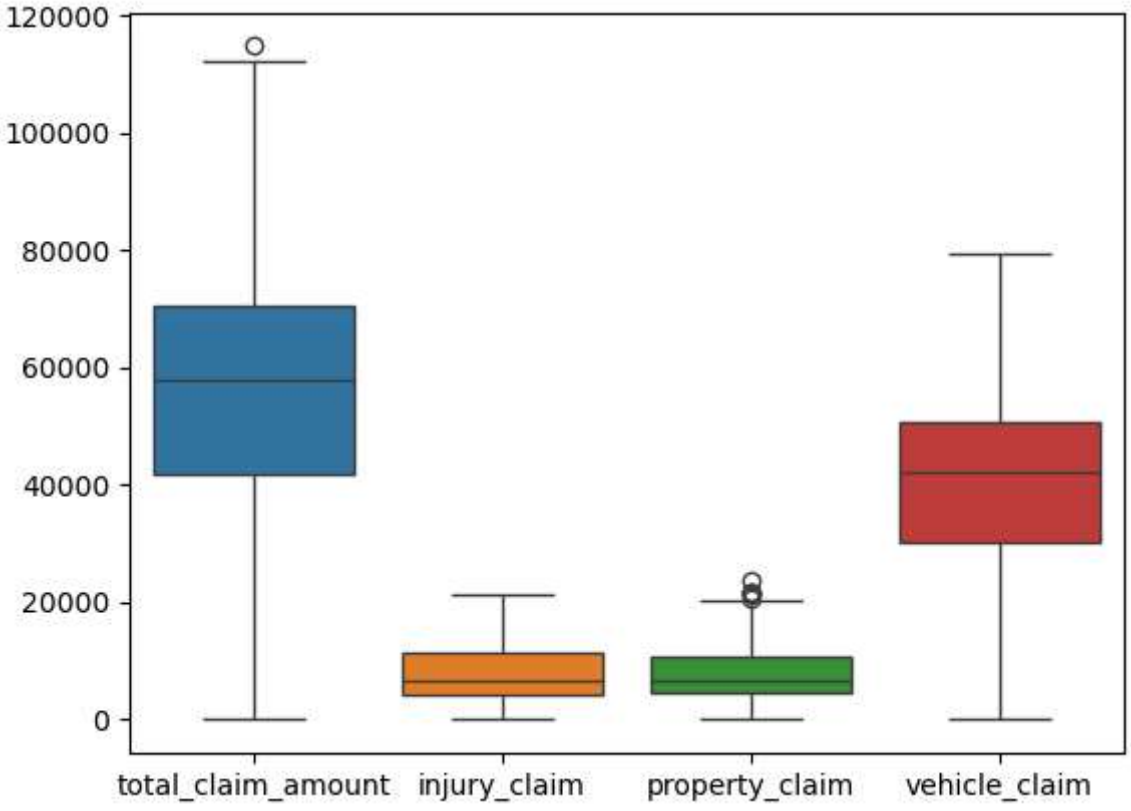
Out[34]:
```
<Axes: >
```



In [ ]:
```
# Observe :- some outliers are available
```

In [32]:
```
sns.boxplot(df[["total_claim_amount","injury_claim","property_claim","vehicle_claim
```

Out[32]: <Axes: >



In [50]:
```
# Observe :- some outliers are available
```

Data Cleaning

In [ ]:
```
# 1>treat wrong Data types
#(umbrella_limit column has wrong data,collision_type,property_damage,police_report
```

In [64]:
```
df.loc[df["umbrella_limit"]==-1000000,"umbrella_limit"]=0
```

In [ ]:
```
# Reason :- Umbrella_limit provides additional liability coverage,
#so the limit should always be zero or positive — never negative.
```

In [94]:
```
df['collision_type'] = df['collision_type'].replace('?', 'Unknown')
```

In [ ]:
```
# Reason:- Replace ? with a meaningful value like "Unknown"
```

In [100…
```
df["property_damage"]=df["property_damage"].replace("?","Unkown")
```

In [ ]:
```
# Reason:- Replace ? with a meaningful value like "Unknown"
```

In [395…
```
df["police_report_available"]=df["police_report_available"].replace("?","Unknown")
```

In [ ]:
```
# Reason:- Replace ? with a meaningful value like "Unknown"
```

In [ ]:
```
# 2>Treat missing value    ("authorities_contacted")
```

In [78]:
```
df["authorities_contacted"].fillna("Uknown",inplace=True)
df["authorities_contacted"].isnull().sum()
```

Out[78]:
```
0
```

In [ ]:
```
# it’s safe, interpretable, and doesn’t guess. that,s why i can not fill with mode
# fill with unknown
```

In [ ]:
```
# 3>Dimension reduction
```

In [295…
```
df.drop(columns=["policy_bind_date","insured_zip","incident_location"],inplace=True
```

In [ ]:
```
# Reason : - almost each rows have unique value  that,s  simply drop them
```

In [ ]:
```
# treat outliers
```

In [ ]:
```
#Retain outliers when they might carry meaningful, real-world insights — especially
```

Data Analysis and Visualization

In [ ]:
```
# Q1. What is the distribution of fraud vs. non-fraud cases?
```

In [159…
```
df["fraud_reported"].value_counts()/len(df["fraud_reported"])*100
```

Out[159…
```
fraud_reported
N    75.3
Y    24.7
Name: count, dtype: float64
```

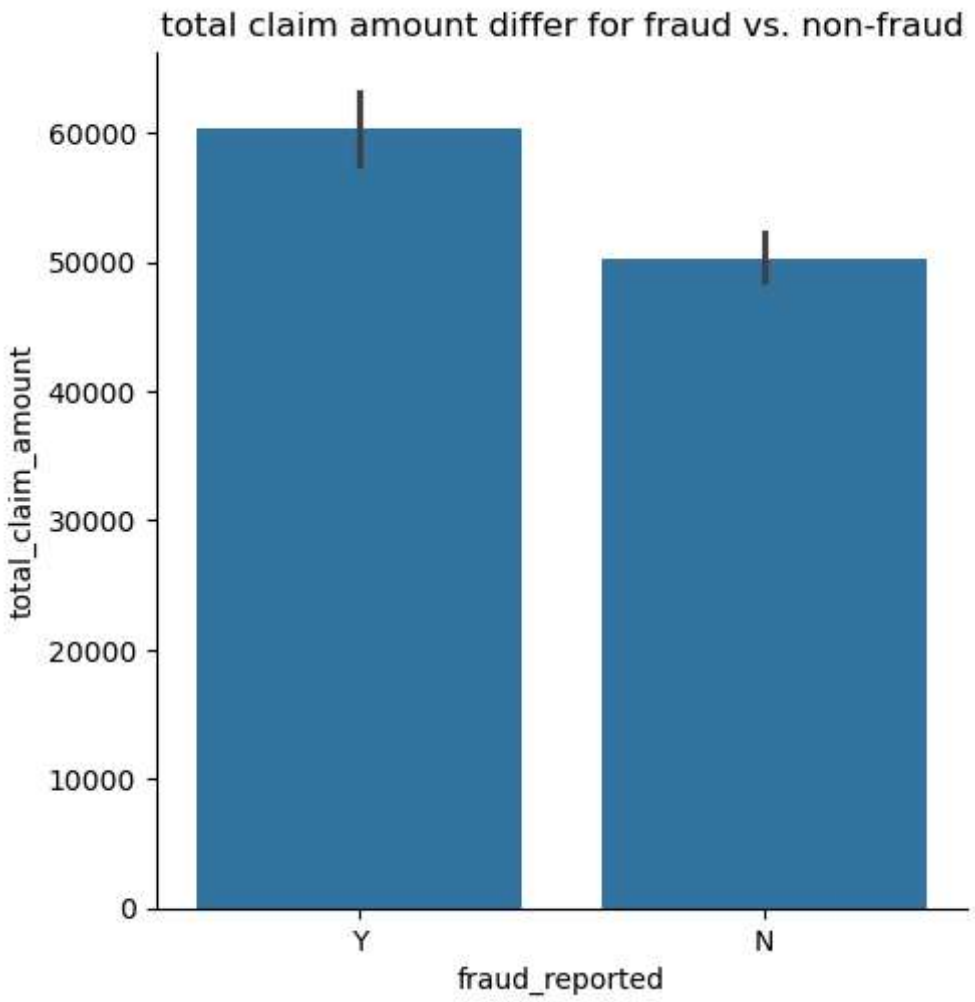
In [383…
```
plt.pie(df["fraud_reported"].value_counts(),labels=["Non-fraud","fraud"],autopct="%
plt.title("distribution of fraud vs. non-fraud")
plt.savefig("froud.jpg")
plt.show()
```

In [ ]:
```
# Q2. How does total claim amount differ for fraud vs. non-fraud?
```

In [187…
```
df.groupby("fraud_reported")["total_claim_amount"].mean()
```

Out[187…
```
fraud_reported
N    50288.605578
Y    60302.105263
Name: total_claim_amount, dtype: float64
```

In [337…
```
sns.catplot(x="fraud_reported",y="total_claim_amount",data=df,kind="bar")
plt.title("total claim amount differ for fraud vs. non-fraud")
plt.savefig("claim_amount.jpg")
plt.show()
```



In [ ]:
```
# answer: -Fraudulent claims tend to have higher total claim amounts on average.
```

In [ ]:
```
#Q3. Which incident types are more likely to be reported as fraud?
```

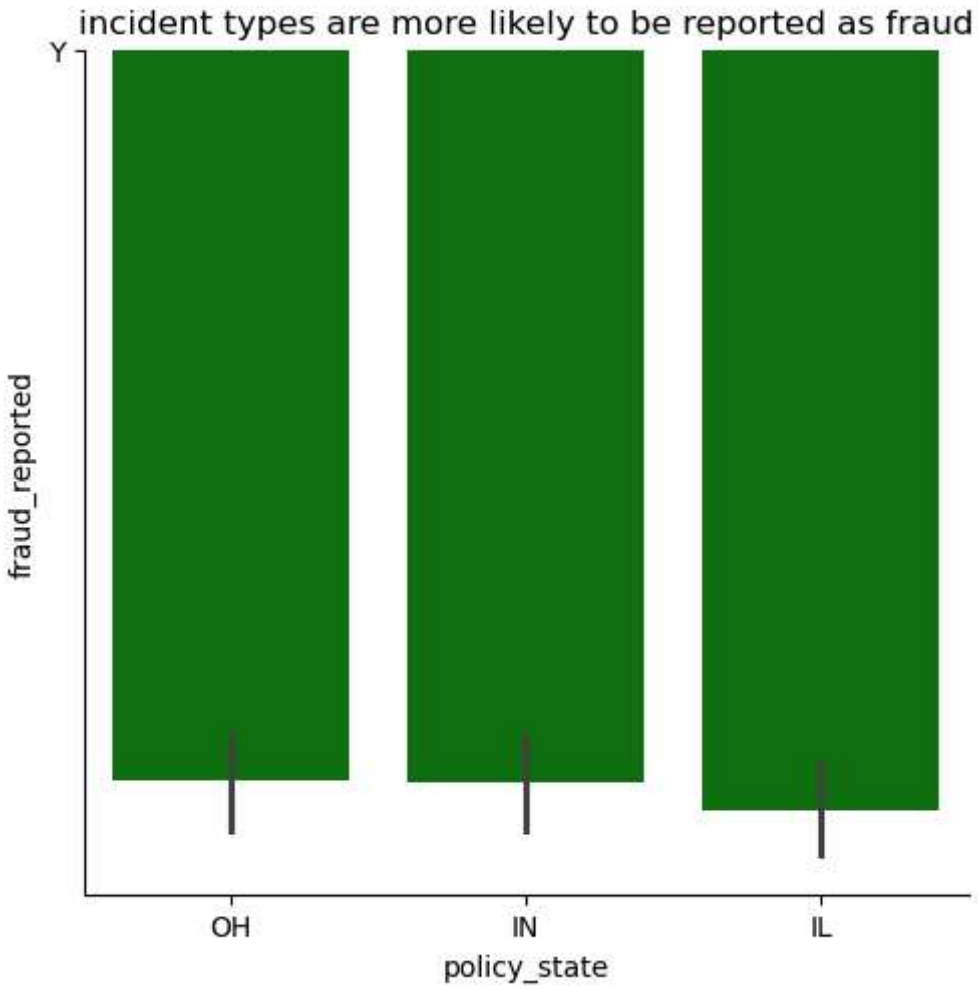
In [ ]:
```
corr=["witnesses",""]
```

In [197…
```
pd.crosstab(df["policy_state"],df["fraud_reported"])
```

Out[197…

policy_state	fraud_reported N	Y
IL	261	77
IN	231	79
OH	261	91

In [385…
```
sns.catplot(y="fraud_reported",x="policy_state",data=df,kind="bar",color="green")
plt.title("incident types are more likely to be reported as fraud")
plt.savefig("froud.jpg")
plt.show()
```



In [ ]:
```
# answer :- IL show higher froud
```

In [ ]:
```
#Q4. Are older or newer cars more involved in fraud claims?
```

In [219…
```
pd.crosstab(df["auto_year"],df["fraud_reported"])
```

Out[219…

fraud_reported	N	Y
auto_year		
1995	43	13
1996	23	14
1997	34	12
1998	33	7
1999	45	10
2000	31	11
2001	33	9
2002	39	10
2003	42	9
2004	23	16
2005	42	12
2006	39	14
2007	34	18
2008	35	10
2009	39	11
2010	43	7
2011	36	17
2012	37	9
2013	34	15
2014	32	12
2015	36	11

In [ ]:
```
# answer:-involved in fraud claims tend to be slightly older than newer not involve
```

In [ ]:
```
#Q5. Does gender influence fraud reporting?
```

In [231…
```
pd.crosstab(df["insured_sex"],df["fraud_reported"])
```

Out[231…

fraud_reported	N	Y
insured_sex		
FEMALE	411	126
MALE	342	121

In [ ]:
```
# answer:- less difference between them
```

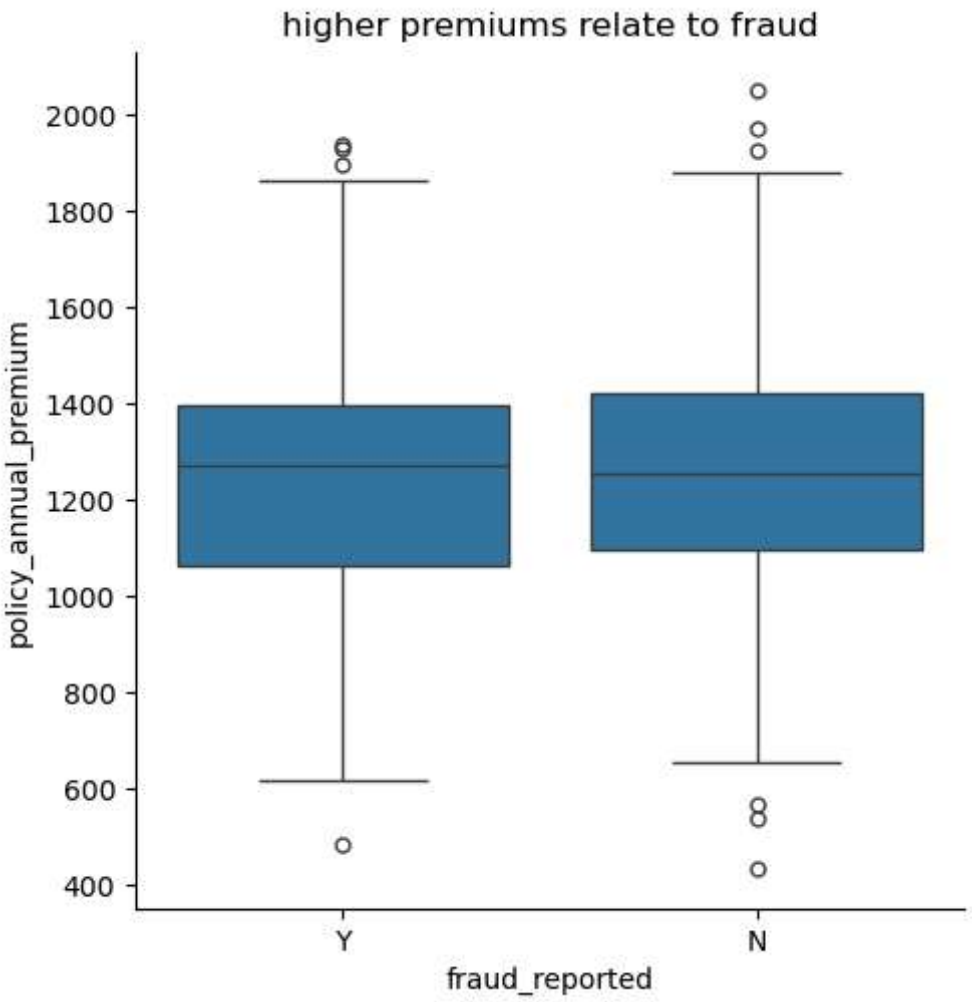
In [ ]:
```
# Q6 which is the higher premiums relate to fraud?
```

In [305…
```
df.groupby("fraud_reported")["policy_annual_premium"].max()
```

Out[305…
```
fraud_reported
N    2047.59
Y    1935.85
Name: policy_annual_premium, dtype: float64
```

In [ ]:
```
# answer= 1935.85
```

In [389…
```
sns.catplot(x="fraud_reported",y="policy_annual_premium",data=df,kind="box")
plt.title("higher premiums relate to fraud")
plt.savefig("froud.jpg")
plt.show()
```



In [ ]:
```
#Q7. Which auto makes are more frequently associated with fraud?
```

In [251…
```
pd.crosstab(df["auto_make"],df["fraud_reported"])
```

Out[251…

auto_make	N	Y
Accura	55	13
Audi	48	21
BMW	52	20
Chevrolet	55	21
Dodge	60	20
Ford	50	22
Honda	41	14
Jeep	56	11
Mercedes	43	22
Nissan	64	14
Saab	62	18
Suburu	61	19
Toyota	57	13
Volkswagen	49	19

In [ ]:
```
# answer:-Ford,mercedes
```

In [ ]:
```
# Q8. Does the presence of a police report impact fraud detection?
```

In [397…
```
pd.crosstab(df["police_report_available"],df["fraud_reported"])
```

Out[397…

police_report_available	N	Y
NO	257	86
Unknown	254	89
YES	242	72

In [ ]:
```
# answer:- less impact on froud detection
```

In [ ]:
```
# Q9 How many witnesses are present in fraud cases?
```

In [285…
```
pd.crosstab(df["witnesses"],df["fraud_reported"]).sum()
```

Out[285…
```
fraud_reported
N    753
Y    247
dtype: int64
```

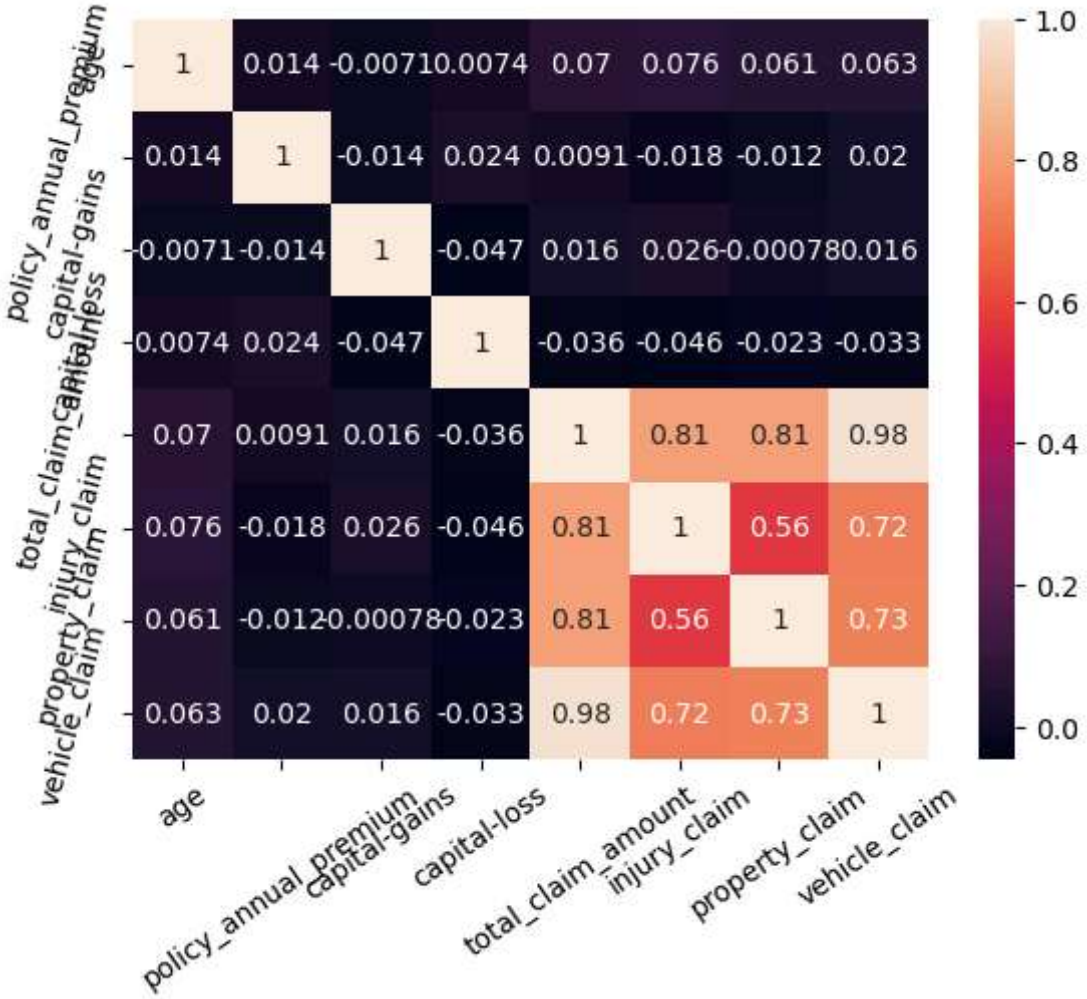
In [ ]:
```
# answer:- 247
```

In [327…
```
corre=df[["age","policy_annual_premium","capital-gains","capital-loss","total_claim
          "property_claim","vehicle_claim"]].corr()
corre
```

Out[327…

	age	policy_annual_premium	capital-gains	capital-loss	total_claim
age	1.000000	0.014404	-0.007075	0.007368	
policy_annual_premium	0.014404	1.000000	-0.013738	0.023547	
capital-gains	-0.007075	-0.013738	1.000000	-0.046904	
capital-loss	0.007368	0.023547	-0.046904	1.000000	-
total_claim_amount	0.069863	0.009094	0.015980	-0.036060	
injury_claim	0.075522	-0.017633	0.025934	-0.046060	
property_claim	0.060898	-0.011654	-0.000779	-0.022863	
vehicle_claim	0.062588	0.020246	0.015836	-0.032665	

In [381…
```
sns.heatmap(corre,annot=True)
plt.xticks(rotation=35)
plt.yticks(rotation=75)
plt.savefig("correlation of conti. variable.jpg")
plt.show()
```



In [ ]:
```
#Strong Positive Correlations:-
#total_claim_amount has a very strong positive correlation with vehicle_claim (0.98
#and injury_claim (0.81

#Weak Correlations:-
#policy_annual_premium, capital-gains, and capital-loss also generally show weak co
# age shows very weak correlations with most other variables,

#Near-Zero or Very Weak Negative Correlations:-
#There are some very weak negative correlations, such as between age and capital-ga
#or age and capital-loss (-0.0074).
```

In [ ]:
```
#  BY (MANISH KUMAR)
```

In [ ]:

In [ ]: