```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import plotly.express as px
import seaborn as sns
import warnings


claim = pd.read_csv('Train_Claim.csv', na_values = "?")
demographics= pd.read_csv('Train_Demographics.csv', na_values = "?")
policy = pd.read_csv('Train_Policy.csv', na_values = "?")
vehicle = pd.read_csv('Train_Vehicle.csv', na_values = "?")
data = pd.read_csv('Traindata_with_Target.csv', na_values = "?")


claim.columns
```

```
Index(['CustomerID', 'DateOfIncident', 'TypeOfIncident',
'TypeOfCollission',
       'SeverityOfIncident', 'AuthoritiesContacted', 'IncidentState',
       'IncidentCity', 'IncidentAddress', 'IncidentTime',
'NumberOfVehicles',
       'PropertyDamage', 'BodilyInjuries', 'Witnesses', 'PoliceReport',
       'AmountOfTotalClaim', 'AmountOfInjuryClaim',
'AmountOfPropertyClaim',
       'AmountOfVehicleDamage'],
      dtype='object')
```

```
claim.shape
```

```
(28836, 19)
```

```
demographics.columns
```

```
Index(['CustomerID', 'InsuredAge', 'InsuredZipCode', 'InsuredGender',
       'InsuredEducationLevel', 'InsuredOccupation', 'InsuredHobbies',
       'CapitalGains', 'CapitalLoss', 'Country'],
      dtype='object')
```

```
demographics.shape
```

```
(28836, 10)
```

```
policy.columns
```

```
Index(['InsurancePolicyNumber', 'CustomerLoyaltyPeriod',
       'DateOfPolicyCoverage', 'InsurancePolicyState',
       'Policy_CombinedSingleLimit', 'Policy_Deductible',
       'PolicyAnnualPremium', 'UmbrellaLimit', 'InsuredRelationship',
       'CustomerID'],
      dtype='object')
```

```
policy.shape
```

```
(28836, 10)
```

```
vehicle.columns
```

```
Index(['CustomerID', 'VehicleAttribute', 'VehicleAttributeDetails'],
dtype='object')
```

```
vehicle.shape
```

```
(115344, 3)
```

```
vehicle.head()
```

| | CustomerID | VehicleAttribute | VehicleAttributeDetails |
|---|---|---|---|
| **0** | Cust20179 | VehicleID | Vehicle8898 |
| **1** | Cust21384 | VehicleModel | Malibu |
| **2** | Cust33335 | VehicleMake | Toyota |
| **3** | Cust27118 | VehicleModel | Neon |
| **4** | Cust13038 | VehicleID | Vehicle30212 |

```
vehicle['VehicleAttribute'].value_counts()
```

```
VehicleID       28836
VehicleModel    28836
VehicleMake     28836
VehicleYOM      28836
Name: VehicleAttribute, dtype: int64
```

```
veh = vehicle.groupby('VehicleAttribute')
```

```
for VehicleAttribute, Vehicle in veh:
  print(VehicleAttribute)
```

```
    VehicleID
    VehicleMake
    VehicleModel
    VehicleYOM
```

```
vehicleId = veh.get_group('VehicleID')
vehicleMake = veh.get_group('VehicleMake')
vehicleModel = veh.get_group('VehicleModel')
vehicleYOM = veh.get_group('VehicleYOM')
```

```
(vehicleId.shape), (vehicleModel.shape), (vehicleMake.shape), (vehicleYOM.shape)
```

```
    ((28836, 3), (28836, 3), (28836, 3), (28836, 3))
```

Renaming columns to avoid confusion

```
vehicleId.rename(columns = {'VehicleAttribute':'VehicleAttribute1', 'VehicleAttr
vehicleModel.rename(columns = {'VehicleAttribute':'VehicleAttribute2', 'VehicleA
vehicleMake.rename(columns = {'VehicleAttribute':'VehicleAttribute3', 'VehicleAt
vehicleYOM.rename(columns = {'VehicleAttribute':'VehicleAttribute4', 'VehicleAtt
```

```
    /usr/local/lib/python3.7/dist-packages/pandas/core/frame.py:5047: SettingWi
    A value is trying to be set on a copy of a slice from a DataFrame

    See the caveats in the documentation: https://pandas.pydata.org/pandas-docs
      errors=errors,
```

```
(vehicleId.columns), (vehicleModel.columns), (vehicleMake.columns), (vehicleYOM.
```

```
    (Index(['CustomerID', 'VehicleAttribute1', 'VehicleAttributeDetails1'],
    dtype='object'),
     Index(['CustomerID', 'VehicleAttribute2', 'VehicleAttributeDetails2'],
    dtype='object'),
     Index(['CustomerID', 'VehicleAttribute3', 'VehicleAttributeDetails3'],
    dtype='object'),
     Index(['CustomerID', 'VehicleAttribute4', 'VehicleAttributeDetails4'],
    dtype='object'))
```

```
data.columns
```

```
    Index(['CustomerID', 'ReportedFraud'], dtype='object')
```

```
data.shape
```

```
(28836, 2)
```

Merging all the Dataframes into single Dataframes using CustomerID

```
df1 = pd.merge(claim, demographics, on = ['CustomerID'])
df2 = pd.merge(df1, policy, on = ['CustomerID'])
df3 = pd.merge(df2, vehicleId, on = ['CustomerID'])
df4 = pd.merge(df3, vehicleModel, on = ['CustomerID'])
df5 = pd.merge(df4, vehicleMake, on = ['CustomerID'])
df6 = pd.merge(df5, vehicleYOM, on = ['CustomerID'])
df = df6 = pd.merge(df6, data, on = ['CustomerID'])
```

```
df.shape
```

```
(28836, 46)
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 28836 entries, 0 to 28835
Data columns (total 46 columns):
 #   Column                       Non-Null Count  Dtype 
---  ------                       --------------  ----- 
 0   CustomerID                   28836 non-null  object
 1   DateOfIncident               28836 non-null  object
 2   TypeOfIncident               28836 non-null  object
 3   TypeOfCollission             23674 non-null  object
 4   SeverityOfIncident           28836 non-null  object
 5   AuthoritiesContacted         28836 non-null  object
 6   IncidentState                28836 non-null  object
 7   IncidentCity                 28836 non-null  object
 8   IncidentAddress              28836 non-null  object
 9   IncidentTime                 28836 non-null  int64 
 10  NumberOfVehicles             28836 non-null  int64 
 11  PropertyDamage               18377 non-null  object
 12  BodilyInjuries               28836 non-null  int64 
 13  Witnesses                    28836 non-null  object
 14  PoliceReport                 19031 non-null  object
 15  AmountOfTotalClaim           28836 non-null  object
 16  AmountOfInjuryClaim          28836 non-null  int64 
 17  AmountOfPropertyClaim        28836 non-null  int64 
 18  AmountOfVehicleDamage        28836 non-null  int64 
 19  InsuredAge                   28836 non-null  int64 
 20  InsuredZipCode               28836 non-null  int64 
 21  InsuredGender                28806 non-null  object
 22  InsuredEducationLevel        28836 non-null  object
```

```
 23  InsuredOccupation           28836 non-null  object
 24  InsuredHobbies              28836 non-null  object
 25  CapitalGains                28836 non-null  int64
 26  CapitalLoss                 28836 non-null  int64
 27  Country                     28834 non-null  object
 28  InsurancePolicyNumber       28836 non-null  int64
 29  CustomerLoyaltyPeriod       28836 non-null  int64
 30  DateOfPolicyCoverage        28836 non-null  object
 31  InsurancePolicyState        28836 non-null  object
 32  Policy_CombinedSingleLimit  28836 non-null  object
 33  Policy_Deductible           28836 non-null  int64
 34  PolicyAnnualPremium         28836 non-null  float64
 35  UmbrellaLimit               28836 non-null  int64
 36  InsuredRelationship         28836 non-null  object
 37  VehicleAttribute1           28836 non-null  object
 38  VehicleAttributeDetails1    28836 non-null  object
 39  VehicleAttribute2           28836 non-null  object
 40  VehicleAttributeDetails2    28836 non-null  object
 41  VehicleAttribute3           28836 non-null  object
 42  VehicleAttributeDetails3    28836 non-null  object
 43  VehicleAttribute4           28836 non-null  object
 44  VehicleAttributeDetails4    28836 non-null  object
 45  ReportedFraud               28836 non-null  object
dtypes: float64(1), int64(14), object(31)
memory usage: 10.3+ MB
```

```
df.head().T
```

| | 0 | 1 | 2 | 3 | |
|---|---|---|---|---|---|
| **CustomerID** | Cust10000 | Cust10001 | Cust10002 | Cust10003 | C |
| **DateOfIncident** | 2015-02-03 | 2015-02-02 | 2015-01-15 | 2015-01-19 | 20 |
| **TypeOfIncident** | Multi-vehicle Collision | Multi-vehicle Collision | Single Vehicle Collision | Single Vehicle Collision | |
| **TypeOfCollission** | Side Collision | Side Collision | Side Collision | Side Collision | |
| **SeverityOfIncident** | Total Loss | Total Loss | Minor Damage | Minor Damage | |
| **AuthoritiesContacted** | Police | Police | Other | Other | |
| **IncidentState** | State7 | State7 | State8 | State9 | |
| **IncidentCity** | City1 | City5 | City6 | City6 | |
| **IncidentAddress** | Location 1311 | Location 1311 | Location 2081 | Location 2081 | |
| **IncidentTime** | 17 | 10 | 22 | 22 | |

| | | | | | |
|---|---|---|---|---|---|
| **NumberOfVehicles** | 3 | 3 | 1 | 1 | |
| **PropertyDamage** | NaN | YES | YES | YES | |
| **BodilyInjuries** | 1 | 2 | 2 | 2 | |
| **Witnesses** | 0 | 1 | 3 | 3 | |
| **PoliceReport** | NaN | YES | NO | NO | |
| **AmountOfTotalClaim** | 65501 | 61382 | 66755 | 66243 | |
| **AmountOfInjuryClaim** | 13417 | 15560 | 11630 | 12003 | |
| **AmountOfPropertyClaim** | 6071 | 5919 | 11630 | 12003 | |
| **AmountOfVehicleDamage** | 46013 | 39903 | 43495 | 42237 | |
| **InsuredAge** | 35 | 36 | 33 | 36 | |
| **InsuredZipCode** | 454776 | 454776 | 603260 | 474848 | |
| **InsuredGender** | MALE | MALE | MALE | MALE | |
| **InsuredEducationLevel** | JD | JD | JD | JD | Hig |
| **InsuredOccupation** | armed-forces | tech-support | armed-forces | armed-forces | m |
| **InsuredHobbies** | movies | cross-fit | polo | polo | |
| **CapitalGains** | 56700 | 70600 | 66400 | 47900 | |
| **CapitalLoss** | -48500 | -48500 | -63700 | -73400 | |
| **Country** | India | India | India | India | |
| **InsurancePolicyNumber** | 119121 | 119122 | 119123 | 119124 | |
| **CustomerLoyaltyPeriod** | 49 | 114 | 167 | 190 | |
| **DateOfPolicyCoverage** | 1998-10-25 | 2000-11-15 | 2001-02-12 | 2005-04-11 | 19 |
| **InsurancePolicyState** | State1 | State1 | State3 | State2 | |
| **Policy_CombinedSingleLimit** | 100/300 | 100/300 | 500/1000 | 500/1000 | |
| **Policy_Deductible** | 1000 | 1000 | 617 | 722 | |
| **PolicyAnnualPremium** | 1632.73 | 1255.19 | 1373.38 | 1337.6 | |
| **UmbrellaLimit** | 0 | 0 | 0 | 0 | |
| **InsuredRelationship** | not-in-family | not-in-family | wife | own-child | u |
| **VehicleAttribute1** | VehicleID | VehicleID | VehicleID | VehicleID | |
| **VehicleAttributeDetails1** | Vehicle26917 | Vehicle15893 | Vehicle5152 | Vehicle37363 | Veh |

| | | | | | |
|---|---|---|---|---|---|
| **VehicleAttribute2** | VehicleModel | VehicleModel | VehicleModel | VehicleModel | Veh |
| **VehicleAttributeDetails2** | A5 | A5 | Jetta | Jetta | |
| **VehicleAttribute3** | VehicleMake | VehicleMake | VehicleMake | VehicleMake | Veh |
| **VehicleAttributeDetails3** | Audi | Audi | Volkswagen | Volkswagen | |

```
df.describe().T
```

| | count | mean | std | min | 25 |
|---|---|---|---|---|---|
| **IncidentTime** | 28836.0 | 11.746047 | 6.170069e+00 | -5.0 | 6.000 |
| **NumberOfVehicles** | 28836.0 | 1.823207 | 9.800987e-01 | 1.0 | 1.000 |
| **BodilyInjuries** | 28836.0 | 0.985782 | 7.847641e-01 | 0.0 | 0.000 |
| **AmountOfInjuryClaim** | 28836.0 | 7337.118428 | 4.427639e+03 | 0.0 | 4743.750 |
| **AmountOfPropertyClaim** | 28836.0 | 7283.870197 | 4.375843e+03 | 0.0 | 4862.000 |
| **AmountOfVehicleDamage** | 28836.0 | 37687.129387 | 1.797705e+04 | 109.0 | 32193.250 |
| **InsuredAge** | 28836.0 | 38.815370 | 7.996377e+00 | 19.0 | 33.000 |
| **InsuredZipCode** | 28836.0 | 502436.579068 | 7.225087e+04 | 430104.0 | 448603.000 |
| **CapitalGains** | 28836.0 | 23066.569566 | 2.763781e+04 | 0.0 | 0.000 |
| **CapitalLoss** | 28836.0 | -24940.612429 | 2.791321e+04 | -111100.0 | -50000.000 |
| **InsurancePolicyNumber** | 28836.0 | 129312.517097 | 1.111406e+04 | 110122.0 | 119698.750 |
| **CustomerLoyaltyPeriod** | 28836.0 | 203.067867 | 9.993295e+01 | 1.0 | 126.000 |
| **Policy_Deductible** | 28836.0 | 1114.282529 | 5.466328e+02 | 500.0 | 622.000 |
| **PolicyAnnualPremium** | 28836.0 | 1255.528382 | 2.230139e+02 | -1.0 | 1122.007 |
| **UmbrellaLimit** | 28836.0 | 983668.034436 | 1.969282e+06 | -1000000.0 | 0.000 |

```
df.nunique()
```

```
CustomerID                   28836
DateOfIncident                  72
TypeOfIncident                   4
TypeOfCollission                 3
SeverityOfIncident               4
AuthoritiesContacted             5
IncidentState                    7
IncidentCity                     7
IncidentAddress               1000
IncidentTime                    25
NumberOfVehicles                 4
PropertyDamage                   2
BodilyInjuries                   3
Witnesses                        5
PoliceReport                     2
AmountOfTotalClaim           21976
AmountOfInjuryClaim          11958
AmountOfPropertyClaim        11785
AmountOfVehicleDamage        20041
InsuredAge                      46
InsuredZipCode                 995
InsuredGender                    2
InsuredEducationLevel            7
InsuredOccupation               14
InsuredHobbies                  20
CapitalGains                   338
CapitalLoss                    354
Country                          1
InsurancePolicyNumber        28836
CustomerLoyaltyPeriod          479
DateOfPolicyCoverage          6779
InsurancePolicyState             3
Policy_CombinedSingleLimit       9
Policy_Deductible             1496
PolicyAnnualPremium          23852
UmbrellaLimit                 7089
InsuredRelationship              6
VehicleAttribute1                1
VehicleAttributeDetails1     28836
VehicleAttribute2                1
VehicleAttributeDetails2        39
VehicleAttribute3                1
VehicleAttributeDetails3        15
VehicleAttribute4                1
VehicleAttributeDetails4        21
ReportedFraud                    2
dtype: int64
```

```
listitem = []
for col in df.columns:
```

```
    listitem.append({
        'column' : col,
        'data type' : df[col].dtype,
        'null total' : df[col].isna().sum(),
        'null pctg' : round(df[col].isna().sum()/len(df[col])*100,2),
        'n_unique' : df[col].nunique(),
    })

pd.DataFrame(listitem)
```

| | column | data type | null total | null pctg | n_unique |
|---|---|---|---|---|---|
| **0** | CustomerID | object | 0 | 0.00 | 28836 |
| **1** | DateOfIncident | object | 0 | 0.00 | 72 |
| **2** | TypeOfIncident | object | 0 | 0.00 | 4 |
| **3** | TypeOfCollission | object | 5162 | 17.90 | 3 |
| **4** | SeverityOfIncident | object | 0 | 0.00 | 4 |
| **5** | AuthoritiesContacted | object | 0 | 0.00 | 5 |
| **6** | IncidentState | object | 0 | 0.00 | 7 |
| **7** | IncidentCity | object | 0 | 0.00 | 7 |
| **8** | IncidentAddress | object | 0 | 0.00 | 1000 |
| **9** | IncidentTime | int64 | 0 | 0.00 | 25 |
| **10** | NumberOfVehicles | int64 | 0 | 0.00 | 4 |
| **11** | PropertyDamage | object | 10459 | 36.27 | 2 |
| **12** | BodilyInjuries | int64 | 0 | 0.00 | 3 |
| **13** | Witnesses | object | 0 | 0.00 | 5 |
| **14** | PoliceReport | object | 9805 | 34.00 | 2 |
| **15** | AmountOfTotalClaim | object | 0 | 0.00 | 21976 |
| **16** | AmountOfInjuryClaim | int64 | 0 | 0.00 | 11958 |
| **17** | AmountOfPropertyClaim | int64 | 0 | 0.00 | 11785 |
| **18** | AmountOfVehicleDamage | int64 | 0 | 0.00 | 20041 |
| **19** | InsuredAge | int64 | 0 | 0.00 | 46 |
| **20** | InsuredZipCode | int64 | 0 | 0.00 | 995 |
| **21** | InsuredGender | object | 30 | 0.10 | 2 |
| **22** | InsuredEducationLevel | object | 0 | 0.00 | 7 |

| | | | | | |
|---|---|---|---|---|---|
| **22** | InsuredEducationLevel | object | 0 | 0.00 | 7 |
| **23** | InsuredOccupation | object | 0 | 0.00 | 14 |
| **24** | InsuredHobbies | object | 0 | 0.00 | 20 |
| **25** | CapitalGains | int64 | 0 | 0.00 | 338 |
| **26** | CapitalLoss | int64 | 0 | 0.00 | 354 |
| **27** | Country | object | 2 | 0.01 | 1 |
| **28** | InsurancePolicyNumber | int64 | 0 | 0.00 | 28836 |
| **29** | CustomerLoyaltyPeriod | int64 | 0 | 0.00 | 479 |
| **30** | DateOfPolicyCoverage | object | 0 | 0.00 | 6779 |
| **31** | InsurancePolicyState | object | 0 | 0.00 | 3 |
| **32** | Policy_CombinedSingleLimit | object | 0 | 0.00 | 9 |
| **33** | Policy_Deductible | int64 | 0 | 0.00 | 1496 |
| **34** | PolicyAnnualPremium | float64 | 0 | 0.00 | 23852 |
| **35** | UmbrellaLimit | int64 | 0 | 0.00 | 7089 |
| **36** | InsuredRelationship | object | 0 | 0.00 | 6 |
| **37** | VehicleAttribute1 | object | 0 | 0.00 | 1 |
| **38** | VehicleAttributeDetails1 | object | 0 | 0.00 | 28836 |
| **39** | VehicleAttribute2 | object | 0 | 0.00 | 1 |
| **40** | VehicleAttributeDetails2 | object | 0 | 0.00 | 39 |
| **41** | VehicleAttribute3 | object | 0 | 0.00 | 1 |
| **42** | VehicleAttributeDetails3 | object | 0 | 0.00 | 15 |
| **43** | VehicleAttribute4 | object | 0 | 0.00 | 1 |
| **44** | VehicleAttributeDetails4 | object | 0 | 0.00 | 21 |
| **45** | ReportedFraud | object | 0 | 0.00 | 2 |

```
for x in df.columns :
  print({x : (df[x].unique())})
```

```
          211, 110, 389, 379, 189,  60, 123, 100,  91,  63, 286, 121, 138,
          184, 178, 157,  61, 154, 180, 302,  76, 156, 158, 160, 242, 102,
          213, 116, 245, 164, 343, 394, 187, 140,  52, 143, 330, 324, 252,
          295, 299, 145, 320, 390, 253, 257, 369,  33,  42, 460, 244, 274,
          196, 200, 195, 204, 217, 185, 107, 266, 267, 210, 370, 296, 201,
           90,  99, 106, 413, 327, 281, 263,  85, 113, 182, 375, 434,   8,
```

```
        62,  98,  30,  80, 260, 367, 312, 351, 356, 364, 396, 357, 352,
       278, 294, 226, 272, 404,  38, 129,  64, 349, 142, 307, 192, 298,
        56, 155,  65, 209, 163, 144, 172, 250, 303, 239, 149, 118, 268,
       332, 159, 124, 238, 313, 193, 431,  89, 199, 306, 400,  79, 119,
        37, 384, 150, 162, 236, 283, 319, 347, 348,  97, 168, 265, 166,
       248, 411, 131, 355, 432, 342, 368,  75,  11,  72, 430, 275, 412,
        86, 282, 232, 231,  84, 325, 148,  78, 305, 151,  94, 270, 229,
        96, 439, 219, 104, 105, 197, 358, 333, 276, 371, 345, 285,  93,
       428, 450, 414, 408, 130,  15, 179, 111, 215, 146, 259,  67, 385,
       454, 141, 254, 188, 478,  73, 128, 338,  51, 206, 126, 177, 249,
       240, 366, 441, 287, 264, 194, 227, 365, 337, 316,  13, 424,  83,
       132, 174, 427,  47, 218,   5, 284, 261, 421,  58, 139,  31, 444,
       339,  82, 456, 464, 221, 207,  50, 103, 117,  88, 108,  92, 183,
        25,  29,  26, 233, 378, 417, 416, 224,  57, 122, 406, 453, 402,
        69, 291, 300,  19, 386, 446, 353,  20, 133, 452, 112,  24, 297,
        46,  23,  66, 377,  55,  22,  71, 398, 361, 279, 309, 258, 423,
       445, 405, 191, 323, 277, 388,  14,  18, 437, 475, 354,  95,  34,
        53,  16, 426, 322, 380, 336, 374,  68, 359,  43, 449,  36, 360,
       443, 459, 468,  77,  39, 376,  48, 362, 372,  32, 315, 448, 397,
       350, 341, 458, 435, 403,   2, 461, 382, 317,   6, 422, 314, 420,
       451, 344, 429, 410, 455, 473,  10, 436, 438, 409,   1, 373,  12,
       415, 393,  35,  21,  40, 462, 466, 433, 457, 387, 472, 308, 442,
        44,   9, 469,  17, 467, 463, 407, 401, 329,   4, 391, 383, 465,
       479, 474, 392,   3,  28, 363, 476, 477, 418, 326, 470])}
{'DateOfPolicyCoverage': array(['1998-10-25', '2000-11-15', '2001-02-12', .
       '2001-11-03', '1998-11-03'], dtype=object)}
{'InsurancePolicyState': array(['State1', 'State3', 'State2'], dtype=object
{'Policy_CombinedSingleLimit': array(['100/300', '500/1000', '250/500', '25
       '100/500', '250/300', '100/1000'], dtype=object)}
{'Policy_Deductible': array([1000,  617,  722, ..., 1104, 1509, 1711])}
{'PolicyAnnualPremium': array([1632.73, 1255.19, 1373.38, ..., 1276.01, 138
{'UmbrellaLimit': array([      0, 4279863, 3921366, ..., 3448735, 3364301,
{'InsuredRelationship': array(['not-in-family', 'wife', 'own-child', 'unmar
       'other-relative'], dtype=object)}
{'VehicleAttribute1': array(['VehicleID'], dtype=object)}
{'VehicleAttributeDetails1': array(['Vehicle26917', 'Vehicle15893', 'Vehicl
       'Vehicle10240', 'Vehicle39163'], dtype=object)}
{'VehicleAttribute2': array(['VehicleModel'], dtype=object)}
{'VehicleAttributeDetails2': array(['A5', 'Jetta', 'CRV', 'C300', 'Passat',
       'Impreza', '93', 'Highlander', 'X5', 'Accord', 'Corolla',
       'Forrestor', 'F150', 'Pathfinder', 'Neon', 'Tahoe', 'Wrangler',
       'A3', 'RSX', 'Malibu', 'E400', 'Legacy', '95', 'Grand Cherokee',
       'Escape', 'Civic', 'Silverado', 'RAM', 'Camry', 'M5', '3 Series',
       'ML350', 'Maxima', 'MDX', 'X6', 'TL'], dtype=object)}
{'VehicleAttribute3': array(['VehicleMake'], dtype=object)}
{'VehicleAttributeDetails3': array(['Audi', 'Volkswagen', 'Toyota', 'Merced
       'Nissan', 'Ford', 'Accura', 'Dodge', 'Honda', 'Chevrolet', 'Jeep',
       'BMW', '???'], dtype=object)}
{'VehicleAttribute4': array(['VehicleYOM'], dtype=object)}
{'VehicleAttributeDetails4': array(['2008', '2006', '1999', '2003', '2010',
       '2004', '2002', '2001', '2005', '1997', '2015', '2012', '2007',
       '2014', '1998', '2009', '1996', '2013'], dtype=object)}
{'ReportedFraud': array(['N', 'Y'], dtype=object)}
```

## Dropping unnecessary features

```
df = df.drop(['CustomerID', 'DateOfIncident', 'IncidentAddress', 'PoliceReport',
```

```
df['DateOfPolicyCoverage'] = pd.to_datetime(df['DateOfPolicyCoverage']).dt.month
```

```
df['IncidentState'] = df['IncidentState'].str.replace('State','')
df['IncidentCity'] = df['IncidentCity'].str.replace('City','')
```

```
df['AmountOfTotalClaim'] = df['AmountOfTotalClaim'].replace('MISSEDDATA',0)
df['Witnesses'] = df['Witnesses'].replace('MISSINGVALUE',0)
```

```
df[['AmountOfTotalClaim', 'Witnesses']] = df[['AmountOfTotalClaim', 'Witnesses']
```

## Handling Missing Values

```
df['TypeOfCollission'].fillna(df['TypeOfCollission'].mode()[0], inplace=True)
df['PropertyDamage'].fillna(df['PropertyDamage'].mode()[0], inplace=True)
df['InsuredGender'].fillna(df['InsuredGender'].mode()[0], inplace=True)
df['Country'].fillna(df['Country'].mode()[0], inplace=True)
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 28836 entries, 0 to 28835
Data columns (total 31 columns):
 #   Column                       Non-Null Count  Dtype  
---  ------                       --------------  -----  
 0   TypeOfIncident               28836 non-null  object 
 1   TypeOfCollission             28836 non-null  object 
 2   SeverityOfIncident           28836 non-null  object 
 3   AuthoritiesContacted         28836 non-null  object 
 4   IncidentState                28836 non-null  object 
 5   IncidentCity                 28836 non-null  object 
 6   IncidentTime                 28836 non-null  int64  
 7   NumberOfVehicles             28836 non-null  int64  
 8   PropertyDamage               28836 non-null  object 
 9   BodilyInjuries               28836 non-null  int64  
 10  Witnesses                    28836 non-null  int64  
 11  AmountOfTotalClaim           28836 non-null  int64  
 12  AmountOfInjuryClaim          28836 non-null  int64  
 13  AmountOfPropertyClaim        28836 non-null  int64  
 14  AmountOfVehicleDamage        28836 non-null  int64  
 15  InsuredAge                   28836 non-null  int64  
 16  InsuredGender                28836 non-null  object 
 17  InsuredEducationLevel        28836 non-null  object 
 18  InsuredOccupation            28836 non-null  object 
 19  InsuredHobbies               28836 non-null  object 
 20  CapitalGains                 28836 non-null  int64  
 21  CapitalLoss                  28836 non-null  int64  
 22  Country                      28836 non-null  object 
 23  CustomerLoyaltyPeriod        28836 non-null  int64  
 24  DateOfPolicyCoverage         28836 non-null  int64  
 25  Policy_CombinedSingleLimit   28836 non-null  object 
 26  Policy_Deductible            28836 non-null  int64  
 27  PolicyAnnualPremium          28836 non-null  float64
 28  UmbrellaLimit                28836 non-null  int64  
 29  InsuredRelationship          28836 non-null  object 
 30  ReportedFraud                28836 non-null  object 
dtypes: float64(1), int64(15), object(15)
memory usage: 7.0+ MB
```

```
df.shape
```

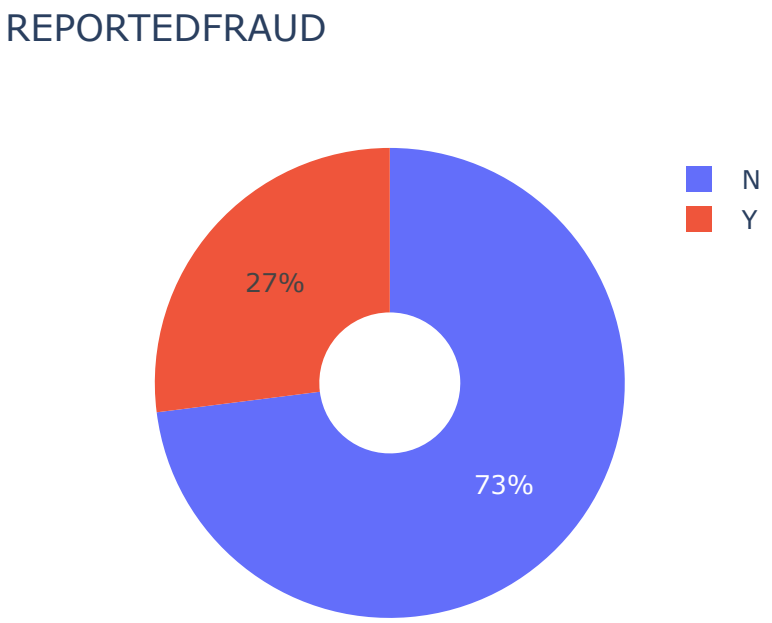
```
(28836, 31)
```

```
X = df.iloc[:, : -1]
Y = df.iloc[:, -1]
```

# Data Analysis

```
# Check the distribution of Target variable to see if it's a case of unbalanced
print(df["ReportedFraud"].value_counts())
names = df["ReportedFraud"].unique()
values = df["ReportedFraud"].value_counts()
fig = px.pie(names = names, values= values, title=col.upper(), width = 400, heig
fig.show()
```

```
N    21051
Y     7785
Name: ReportedFraud, dtype: int64
```




```
# Converting the object into Category
cat_cols = df.select_dtypes(include='object').columns
for col in cat_cols:
    df[col] = df[col].astype('category')
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 28836 entries, 0 to 28835
Data columns (total 31 columns):
 #   Column                       Non-Null Count  Dtype   
---  ------                       --------------  -----   
 0   TypeOfIncident               28836 non-null  category
 1   TypeOfCollission             28836 non-null  category
 2   SeverityOfIncident           28836 non-null  category
 3   AuthoritiesContacted         28836 non-null  category
 4   IncidentState                28836 non-null  category
 5   IncidentCity                 28836 non-null  category
 6   IncidentTime                 28836 non-null  int64   
 7   NumberOfVehicles             28836 non-null  int64   
 8   PropertyDamage               28836 non-null  category
 9   BodilyInjuries               28836 non-null  int64   
 10  Witnesses                    28836 non-null  int64   
 11  AmountOfTotalClaim           28836 non-null  int64   
 12  AmountOfInjuryClaim          28836 non-null  int64   
 13  AmountOfPropertyClaim        28836 non-null  int64   
 14  AmountOfVehicleDamage        28836 non-null  int64   
 15  InsuredAge                   28836 non-null  int64   
 16  InsuredGender                28836 non-null  category
 17  InsuredEducationLevel        28836 non-null  category
 18  InsuredOccupation            28836 non-null  category
 19  InsuredHobbies               28836 non-null  category
 20  CapitalGains                 28836 non-null  int64   
 21  CapitalLoss                  28836 non-null  int64   
 22  Country                      28836 non-null  category
 23  CustomerLoyaltyPeriod        28836 non-null  int64   
 24  DateOfPolicyCoverage         28836 non-null  int64   
 25  Policy_CombinedSingleLimit   28836 non-null  category
 26  Policy_Deductible            28836 non-null  int64   
 27  PolicyAnnualPremium          28836 non-null  float64 
 28  UmbrellaLimit                28836 non-null  int64   
 29  InsuredRelationship          28836 non-null  category
 30  ReportedFraud                28836 non-null  category
dtypes: category(15), float64(1), int64(15)
memory usage: 4.2 MB
```

```
cat_cols = ['TypeOfIncident', 'TypeOfCollission', 'SeverityOfIncident', 'Authori
```

```
num_cols = ['IncidentTime', 'NumberOfVehicles', 'BodilyInjuries', 'Witnesses', '
```

Train, Test, Split

```
from sklearn.model_selection import train_test_split
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.30, random
```

```
print(X_train.shape)
print(X_test.shape)
print(Y_train.shape)
print(Y_test.shape)
```

```
(20185, 30)
(8651, 30)
(20185,)
(8651,)
```

```
Y_train.value_counts(normalize=True)*100
```

```
N    72.836265
Y    27.163735
Name: ReportedFraud, dtype: float64
```

```
Y_test.value_counts(normalize=True)*100
```

```
N    73.390359
Y    26.609641
Name: ReportedFraud, dtype: float64
```

```
df_cat_train = X_train[cat_cols]
df_cat_test = X_test[cat_cols]
print(df_cat_train.shape)
print(df_cat_test.shape)
```

```
(20185, 14)
(8651, 14)
```

```
df_num_train = X_train[num_cols]
df_num_test = X_test[num_cols]
print(df_num_train.shape)
print(df_num_test.shape)
```

```
(20185, 16)
(8651, 16)
```

Converting Categorical data to numeric data

```
from sklearn.preprocessing import OneHotEncoder
ohe = OneHotEncoder(handle_unknown='ignore')
ohe.fit(df_cat_train)

df_cat_train_ohe = ohe.transform(df_cat_train).toarray()
df_cat_test_ohe = ohe.transform(df_cat_test).toarray()


print(df_cat_train_ohe.shape)
print(df_cat_test_ohe.shape)
```

```
    (20185, 91)
    (8651, 91)
```

```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
scaler.fit(df_num_train)
df_num_train_ss = scaler.transform(df_num_train)
df_num_test_ss = scaler.transform(df_num_test)


print(df_num_train_ss.shape)
print(df_num_test_ss.shape)
```

```
    (20185, 16)
    (8651, 16)
```

```
X_train = np.concatenate([df_cat_train_ohe, df_num_train_ss], axis=1)
X_test = np.concatenate([df_cat_test_ohe, df_num_test_ss], axis=1)
```

## Logistic Regression

```
from sklearn.linear_model import LogisticRegression
lr = LogisticRegression()
lr.fit(X_train, Y_train)
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/linear_model/_logistic.py:81

lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regr

LogisticRegression()
```

```
lr_train_pred = lr.predict(X_train)
lr_train_predScore = lr.score(X_train, Y_train)
lr_train_predScore
```

```
0.8396333911320287
```

```
lr_test_pred = lr.predict(X_test)
lr_test_predScore = lr.score(X_test, Y_test)
lr_test_predScore
```

```
0.8363195006357647
```

## RandomForestClassifier

```
from sklearn.ensemble import RandomForestClassifier
rfc = RandomForestClassifier(n_estimators = )
rfc.fit(X_train, Y_train)
```

```
RandomForestClassifier()
```

```
rfc_train_pred = rfc.predict(X_train)
rfc_train_predScore = rfc.score(X_train, Y_train)
rfc_train_predScore
```

```
0.99990091652217
```

```
rfc_test_pred = rfc.predict(X_test)
rfc_test_predScore = rfc.score(X_test, Y_test)
rfc_test_predScore
```

```
0.9204716217778292
```

## KNeighborsClassifier

```
from sklearn.neighbors import KNeighborsClassifier
knc = KNeighborsClassifier()
knc.fit(X_train, Y_train)
```

```
KNeighborsClassifier()
```

```
knc_train_pred = knc.predict(X_train)
knc_train_predScore = knc.score(X_train, Y_train)
knc_train_predScore
```

```
0.9431260837255387
```

```
knc_test_pred = knc.predict(X_test)
knc_test_predScore = knc.score(X_test, Y_test)
knc_test_predScore
```

```
0.9391977806033984
```

## DecisionTreeClassifier

```
from sklearn.tree import DecisionTreeClassifier
dtc = DecisionTreeClassifier()
dtc.fit(X_train, Y_train)
```

```
DecisionTreeClassifier()
```

```
dtc_train_pred = dtc.predict(X_train)
dtc_train_predScore = dtc.score(X_train, Y_train)
dtc_train_predScore
```

```
1.0
```

```
dtc_test_pred = dtc.predict(X_test)
dtc_test_predScore = dtc.score(X_test, Y_test)
dtc_test_predScore
```

```
0.8343544098948098
```

## SVC

```
from sklearn.svm import SVC
svc = SVC(kernel = 'rbf')
svc.fit(X_train, Y_train)
```

```
SVC()
```

```
svc_train_pred = svc.predict(X_train)
svc_train_predScore = svc.score(X_train, Y_train)
svc_train_predScore
```

```
0.9415407480802577
```

```
svc_test_pred = svc.predict(X_test)
svc_test_predScore = svc.score(X_test, Y_test)
svc_test_predScore
```

```
0.9312218240665819
```

## XGBClassifier

```
from xgboost import XGBClassifier
xgb = XGBClassifier(max_depth = 8)
xgb.fit(X_train, Y_train)
```

```
XGBClassifier(max_depth=8)
```

```
xgb_train_pred = xgb.predict(X_train)
xgb_train_predScore = xgb.score(X_train, Y_train)
xgb_train_predScore
```

```
0.9466930889274213
```

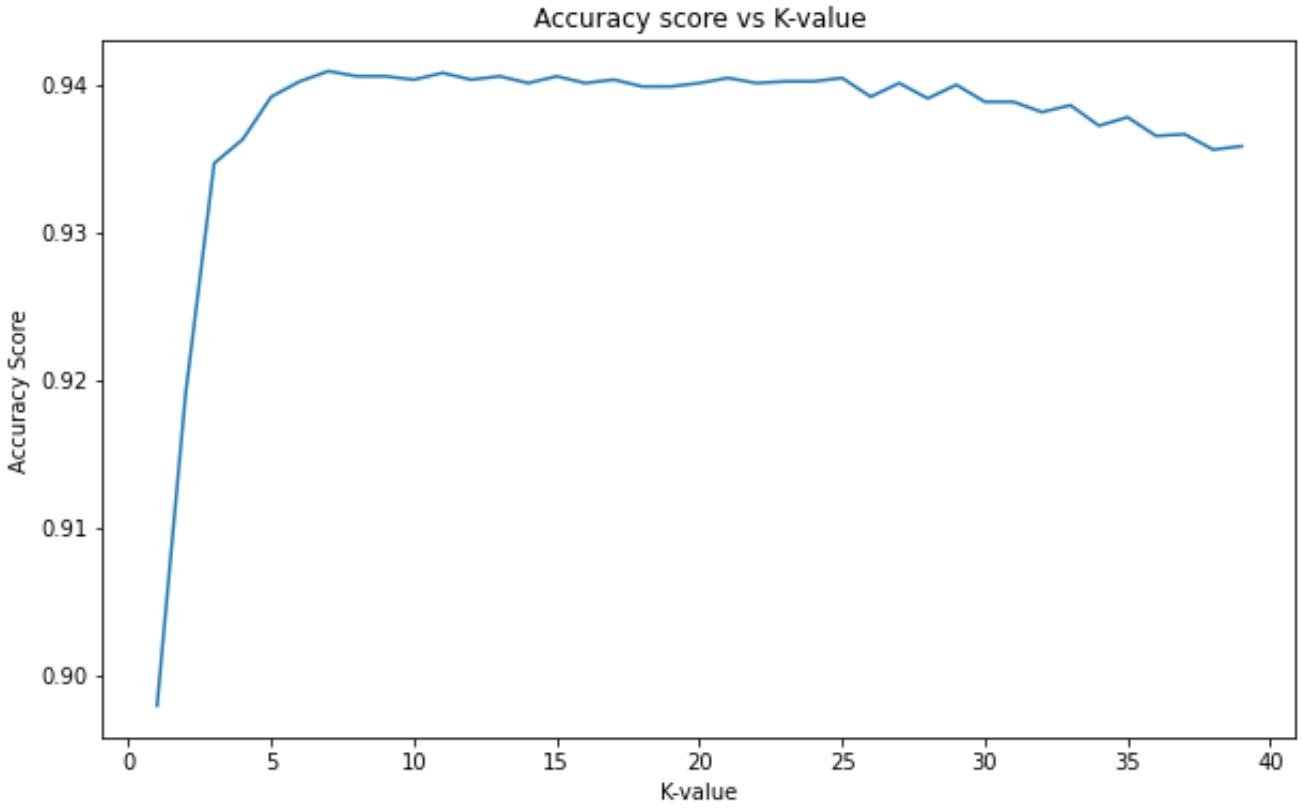
```
xgb_test_pred = xgb.predict(X_test)
xgb_test_predScore = xgb.score(X_test, Y_test)
xgb_test_predScore
```

```
0.9151543174199515
```

```
from sklearn import metrics
acc_score = []
for i in range(1, 40):
  kclassifier = KNeighborsClassifier(n_neighbors = i)
  kclassifier.fit(X_train, Y_train)
  Y_predk = kclassifier.predict(X_test)
  acc_score.append(metrics.accuracy_score(Y_test, Y_predk))
```

```
plt.figure(figsize = (10, 6))
plt.plot(range(1, 40), acc_score)
plt.title('Accuracy score vs K-value')
plt.xlabel('K-value')
plt.ylabel('Accuracy Score')
print("Maximum Accuracy:-",max(acc_score) * 100,"at K =",acc_score.index(max(acc
```

```
Maximum Accuracy:- 94.09316841983586 at K = 6
```



Colab paid products - Cancel contracts here