Predictive Modeling

Haresh P Tayade
PGP-DSBA
Online Sept’2021
Date-29/03/2022

Logo
Name

Problem : 1

Linear Regression

Problem Statement: You are a part of an investing firm and your work is to do research about these 759 firms. You are provided with the dataset containing the sales and other attributes of these 759 firms. Predict the sales of these firms on the bases of the details given in the dataset so as to help your company in investing consciously. Also, provide them with 5 attributes that are most important.

Data Dictionary for Firm_level_data:

1. sales: Sales (in millions of dollars).

2. capital: Net stock of property, plant, and equipment.

3. patents: Granted patents.

4. randd: R&D stock (in millions of dollars).

5. employment: Employment (in 1000s).

6. sp500: Membership of firms in the S&P 500 index. S&P, is a stock market index that measures the stock performance of

500 large companies listed on stock exchanges in the United States

7. tobinq: Tobin's q (also known as q ratio and Kaldor's v) is the ratio between a physical asset's market value and its replacement value.

8. value: Stock market value.

9. institutions: Proportion of stock owned by institutions.

Importing important Libraries

Question 1.1

Read the data and do exploratory data analysis. Describe the data briefly. (Check the null values, data

types, shape, EDA). Perform Univariate and Bivariate Analysis.

```
df.head()
```

	Unnamed: 0	sales	capital	patents	randd	employment	sp500	tobinq	value	institutions
0	0	826.995050	161.603986	10	382.078247	2.306000	no	11.049511	1625.453755	80.27
1	1	407.753973	122.101012	2	0.000000	1.860000	no	0.844187	243.117082	59.02
2	2	8407.845588	6221.144614	138	3296.700439	49.659005	yes	5.205257	25865.233800	47.70
3	3	451.000010	266.899987	1	83.540161	3.071000	no	0.305221	63.024630	26.88
4	4	174.927981	140.124004	2	14.233637	1.947000	no	1.063300	67.406408	49.46

```
df.tail()
```

	Unnamed: 0	sales	capital	patents	randd	employment	sp500	tobinq	value	institutions
754	754	1253.900196	708.299935	32	412.936157	22.100002	yes	0.697454	267.119487	33.50
755	755	171.821025	73.666008	1	0.037735	1.684000	no	NaN	228.475701	46.41
756	756	202.726967	123.926991	13	74.861099	1.460000	no	5.229723	580.430741	42.25
757	757	785.687944	138.780992	6	0.621750	2.900000	yes	1.625398	309.938651	61.39
758	758	22.701999	14.244999	5	18.574360	0.197000	no	2.213070	18.940140	7.50

Head and Tail of dataframe

Shape ,info,datatypes of dataframe

```
df.shape
```

```
(759, 10)
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 759 entries, 0 to 758
Data columns (total 10 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   Unnamed: 0    759 non-null    int64  
 1   sales         759 non-null    float64
 2   capital       759 non-null    float64
 3   patents       759 non-null    int64  
 4   randd         759 non-null    float64
 5   employment    759 non-null    float64
 6   sp500         759 non-null    object 
 7   tobinq        738 non-null    float64
 8   value         759 non-null    float64
 9   institutions  759 non-null    float64
dtypes: float64(7), int64(2), object(1)
memory usage: 59.4+ KB
```

```
df.dtypes
```

```
Unnamed: 0        int64
sales           float64
capital         float64
patents           int64
randd           float64
employment      float64
sp500            object
tobinq          float64
value           float64
institutions    float64
dtype: object
```

	count	mean	std	min	25%	50%	75%	max
Unnamed: 0	759.0	379.000000	219.248717	0.000000	189.500000	379.000000	568.500000	758.000000
sales	759.0	2689.705158	8722.060124	0.138000	122.920000	448.577082	1822.547366	135696.788200
capital	759.0	1977.747498	6466.704896	0.057000	52.650501	202.179023	1075.790020	93625.200560
patents	759.0	25.831357	97.259577	0.000000	1.000000	3.000000	11.500000	1220.000000
randd	759.0	439.938074	2007.397588	0.000000	4.628262	36.864136	143.253403	30425.255860
employment	759.0	14.164519	43.321443	0.006000	0.927500	2.924000	10.050001	710.799925
tobinq	738.0	2.794910	3.366591	0.119001	1.018783	1.680303	3.139309	20.000000
value	759.0	2732.734750	7071.072362	1.971053	103.593946	410.793529	2054.160386	95191.591160
institutions	759.0	43.020540	21.685586	0.000000	25.395000	44.110000	60.510000	90.150000

Description of dataframe

Checking for duplicates

```
df.duplicated().sum()
```

```
0
```

Checking for null values

```
Unnamed: 0       0
sales            0
capital          0
patents          0
randd            0
employment       0
sp500            0
tobinq          21
value            0
institutions     0
dtype: int64
```

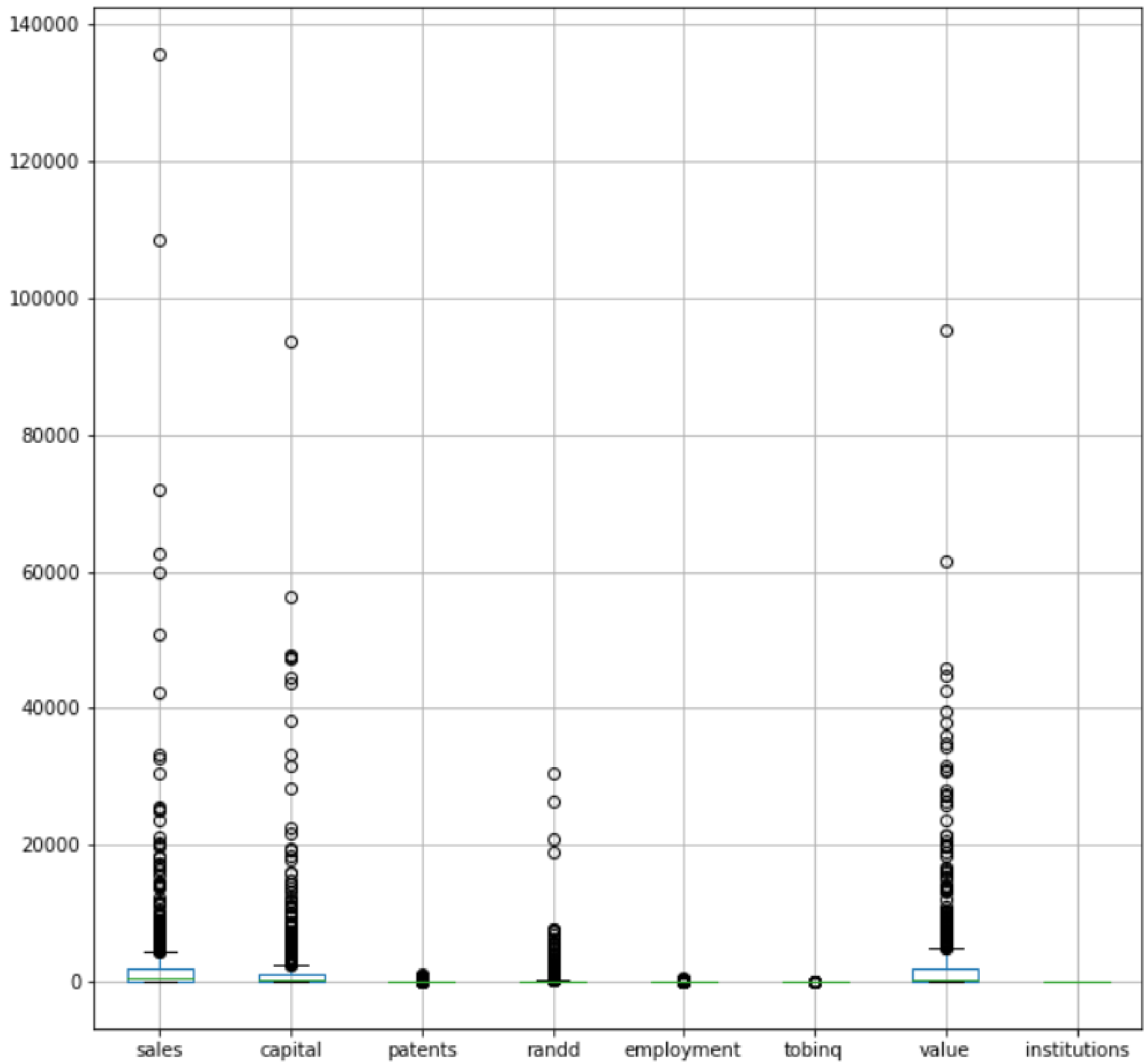
Observations –

- **The above dataset contains 759 rows & 10 columns.**
- **As the 1st column is not used so need to remove that in our entire work.**
    - **The variable 'sp500' is of object datatype, whereas the rest of variable is of numerical (integer & float)datatype.**
- **There are 21 null values in tobniq variable. hence we need to further investigate to treat them.**
- **The summary table shows mean, standard deviation, minimum & maximum values, etc. for all the variables.**

Univariate Analysis

1) **Outlier Identification**

```
<AxesSubplot:>
```

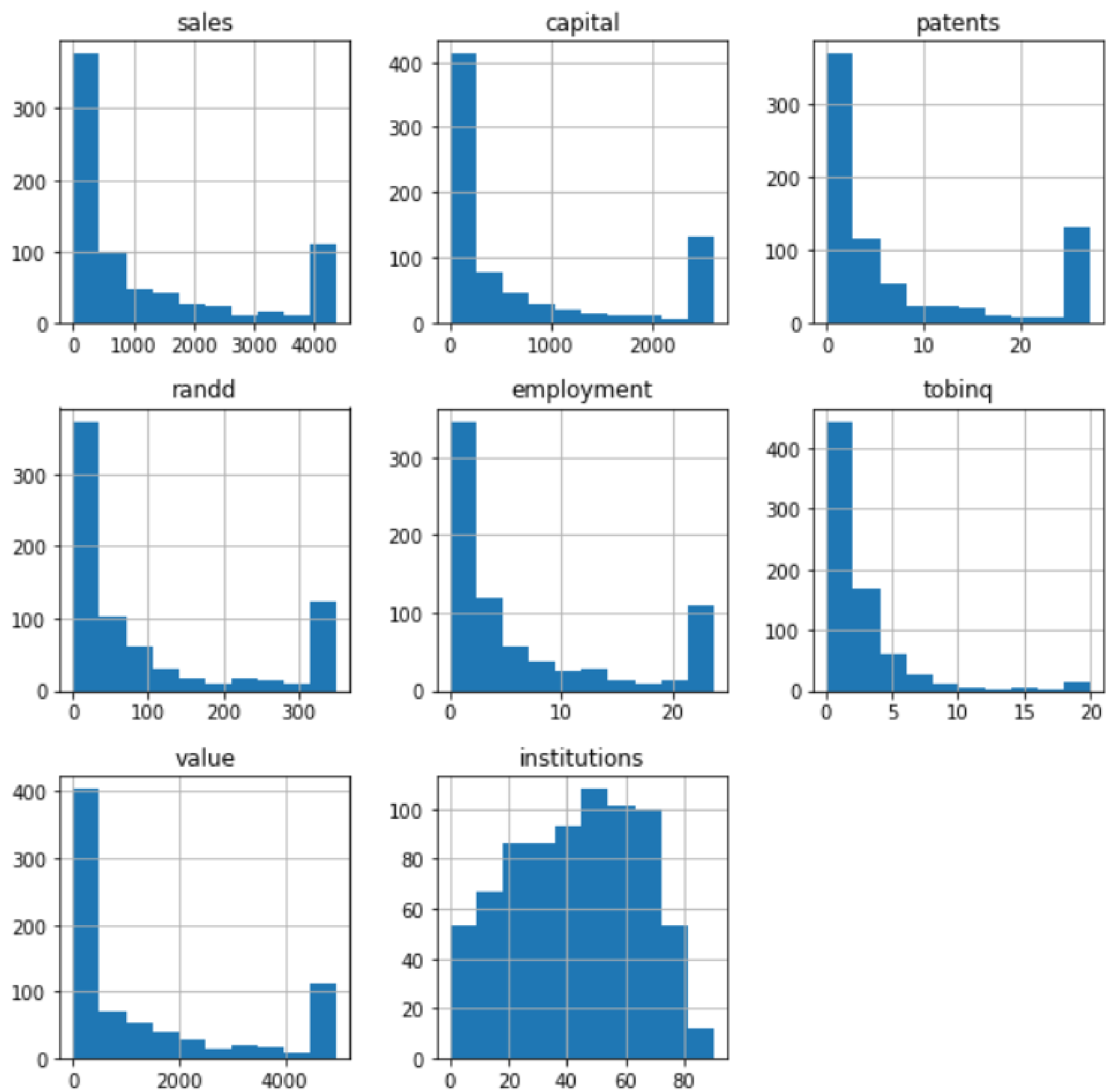


from the figure above ,we can say that

- **All variable contain outlier and since we don't know the reason behind those , so we need to remove those outliers**

- **Treating outliers sometimes result in the models having better performance but the models loose out generalization.**

2)Distribution check



HISTPLOT *FOR distribution check for all variable*

Skew values for all variable

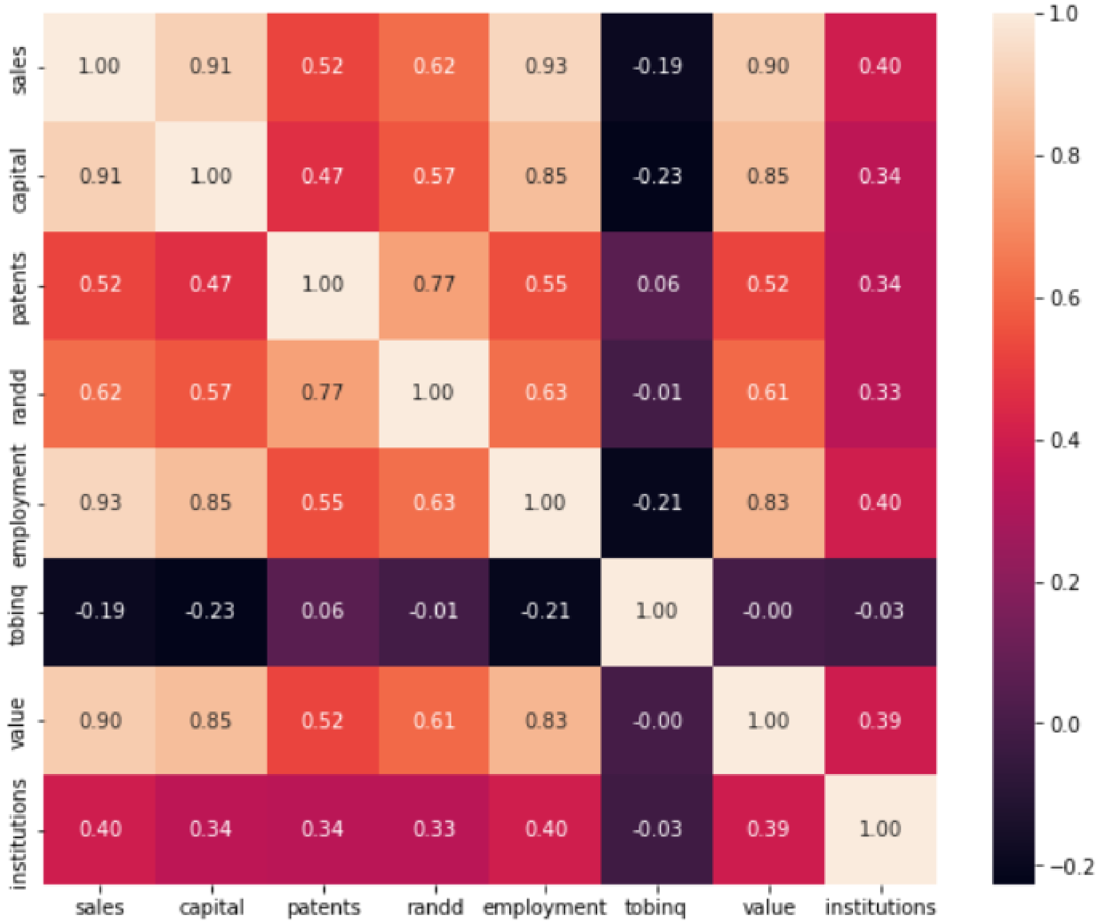
```
institutions   -0.168071
patents         1.162219
randd           1.162978
employment      1.186553
sales           1.189942
capital         1.190265
value           1.195849
tobinq          3.285773
dtype: float64
```

From fig and skew values above, we can say that

- **Since, the Skewness value of variables 'institutions' is between -0.5 and +0.5 they show symmetric distribution.**
- **Since the Skewness value of variable 'patents', 'randd', 'employment', 'sales', 'capital', 'value' is between +1 to +1.2.**
- **Since the Skewness value of variable ' tobniq' is greater that +1.2 ,it showed highly skewed distribution.**

Multivariate analysis

1) Heatmap to study correlation between variables.

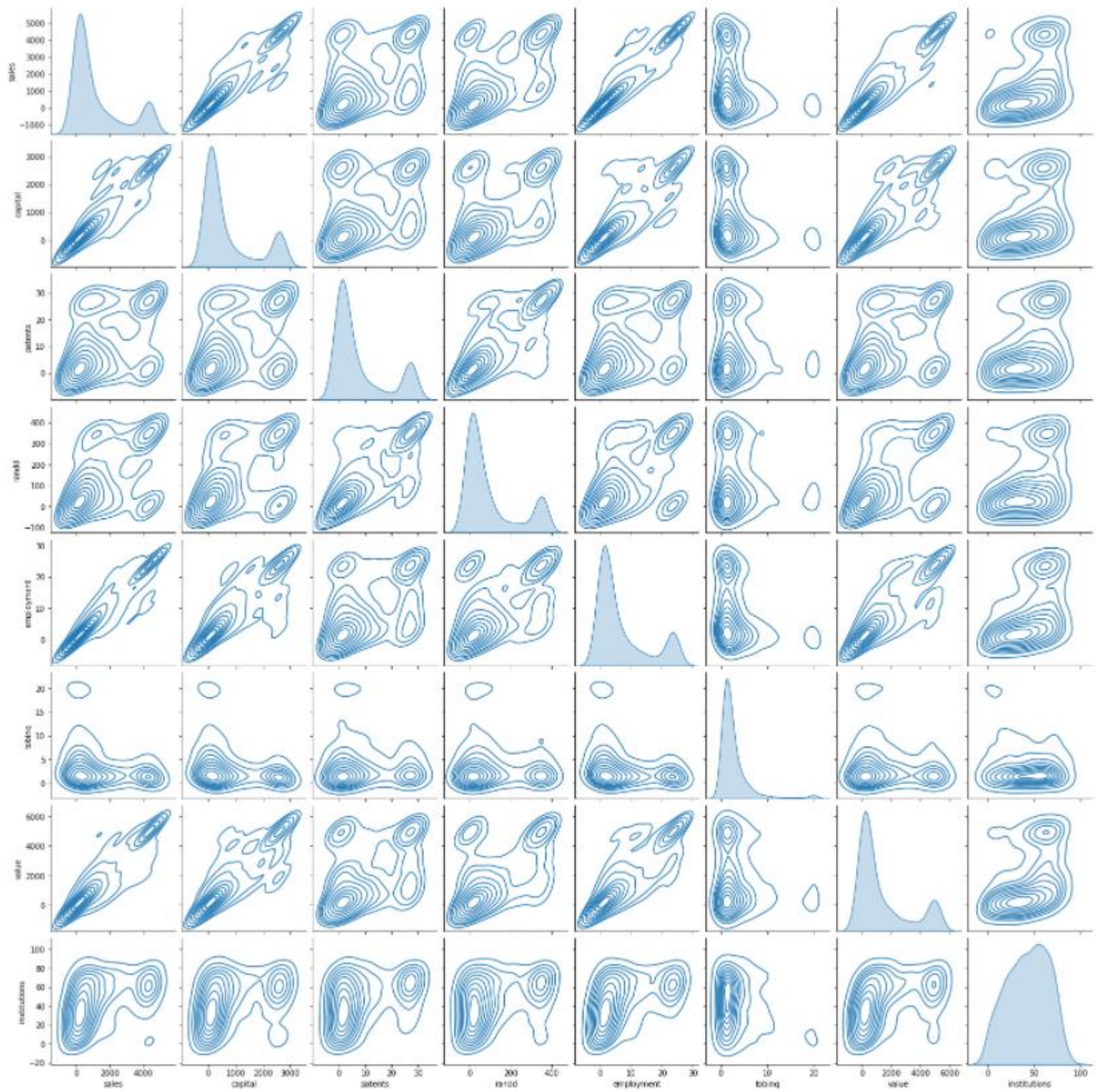


From the fig above we can say that

There is a strong correlation between the variables

(sales – capital,sales-employment,sales-value)(i.e>0.90)

<seaborn.axisgrid.PairGrid at 0x2dca405b8e0>



Pairplot for all variable combination

From the fig above , we can say that

- All the correlation can be viewed as in the pairplot as the datapoints are closely packed to each other in above combination of variables.
- some variables do have correlation, but is very weak in strength which is evident in the graph

Question 1.2

Impute null values if present.Do you think scaling is necessary in this case?

Answer

Since, mean takes all the values of the dataset into consideration , we will impute the null values with the respective mean value of the variable.

Null value before imputing

```
sales            0
capital          0
patents          0
randd            0
employment       0
sp500            0
tobinq          21
value            0
institutions     0
dtype: int64
```

Null value after imputing

```
sales           0
capital         0
patents         0
randd           0
employment      0
sp500           0
tobinq          0
value           0
institutions    0
dtype: int64
```

Scaling

If scaling is not applied VIF variation inflation factor values are high that indicates the presence of multi-collinearity .these values are calculated after building

the linear Regression model also to understand the multicollinearity in the model .

Therefore scaling is much need for this data.

Question 1.3

Encode the data (having string values) for modelling. data split:split the data into test and train (70:30). Apply Linear Regression Performance Metrics: check the performance on predictions on train and test using Rsquare,RMSE

Answer

```
sales            float64
capital          float64
patents          float64
randd            float64
employment       float64
sp500             object
tobinq           float64
value            float64
institutions     float64
dtype: object
```

From above fig we can say that

All the variables are numerical except sp500 which is object

Now we need to convert the sp500 into numerical datatype

```
sales           float64
capital         float64
patents         float64
randd           float64
employment      float64
sp500           float64
tobinq          float64
value           float64
institutions    float64
dtype: object
```

From above fig we can say

That we have already converted categorical variable to numerical variable.

Now splitting the data into train and test

```
X = df.drop('sales', axis=1)

y = df[['sales']]

print(X.head())
print('\n')
print(y.head())
```

```
       capital  patents       randd  employment  sp500     tobinq  \
0   161.603986    10.00  351.191114    2.306000    0.0  11.049511
1   122.101012     2.00    0.000000    1.860000    0.0   0.844187
2  2610.499299    27.25  351.191114   23.733752    1.0   5.205257
3   266.899987     1.00   83.540161    3.071000    0.0   0.305221
4   140.124004     2.00   14.233637    1.947000    0.0   1.063300

         value  institutions
0  1625.453755         80.27
1   243.117082         59.02
2  4980.010044         47.70
3    63.024630         26.88
4    67.406408         49.46


         sales
0   826.995050
1   407.753973
2  4371.988416
3   451.000010
4   174.927981
```

We can perform linear regression using following method

1)Linear regression using sklearn

We know the variable ‘sales’ is target varaiable and we split the data into (70:30) (Train:Test)

Once we build the model we find the coefficients for all the variables and intercept for the linear equation.

```
The coefficient for capital is 0.42847423102599097
The coefficient for patents is -4.707201647817153
The coefficient for randd is 0.5938441977302858
The coefficient for employment is 79.64381157697416
The coefficient for sp500 is 169.71110467472752
The coefficient for tobinq is -16.480262742396423
The coefficient for value is 0.23144250962508106
The coefficient for institutions is 0.0893484177453614
```

The intercept for our model is 28.811904631502102

R^2 for training data 0.9354491655587379

R^2 for testing data 0.923116101085882

RMSE 402.40387301430326

Observations:

We see the variable sp500 is the most important or governing parameter.

The model score for training and testing data is almost same, hence the model is good fitted model.

The accuracy of the model is appproximately is **67% .**

2) Linear regression using stats model

First we concatenate X and y into a single dataframes

Once we build the model we find the coeffiecient for all the variables and intercept for the linear equation.

```
Intercept        28.811905
capital           0.428474
patents          -4.707202
randd             0.593844
employment       79.643812
sp500           169.711105
tobinq          -16.480263
value             0.231443
institutions      0.089348
dtype: float64
```

OLS Test Result:

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                  sales   R-squared:                       0.935
Model:                            OLS   Adj. R-squared:                  0.934
Method:                 Least Squares   F-statistic:                     945.6
Date:                Mon, 28 Mar 2022   Prob (F-statistic):          6.02e-305
Time:                        20:36:50   Log-Likelihood:                -3929.5
No. Observations:                 531   AIC:                             7877.
Df Residuals:                     522   BIC:                             7915.
Df Model:                           8
Covariance Type:            nonrobust
================================================================================
                   coef    std err          t      P>|t|      [0.025      0.975]
--------------------------------------------------------------------------------
Intercept       28.8119     44.084      0.654      0.514     -57.791     115.415
capital          0.4285      0.041     10.547      0.000       0.349       0.508
patents         -4.7072      2.806     -1.677      0.094     -10.220       0.806
randd            0.5938      0.233      2.547      0.011       0.136       1.052
employment      79.6438      4.751     16.763      0.000      70.310      88.978
sp500          169.7111     66.709      2.544      0.011      38.661     300.761
tobinq         -16.4803      6.122     -2.692      0.007     -28.508      -4.453
value            0.2314      0.025      9.405      0.000       0.183       0.280
institutions     0.0893      0.905      0.099      0.921      -1.688       1.867
==============================================================================
Omnibus:                      183.987   Durbin-Watson:                   1.955
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             1261.544
Skew:                           1.341   Prob(JB):                    1.15e-274
Kurtosis:                      10.059   Cond. No.                     9.82e+03
==============================================================================
```

Note:

Standard error assume that the covariance matrix of the errors is correctly specified.

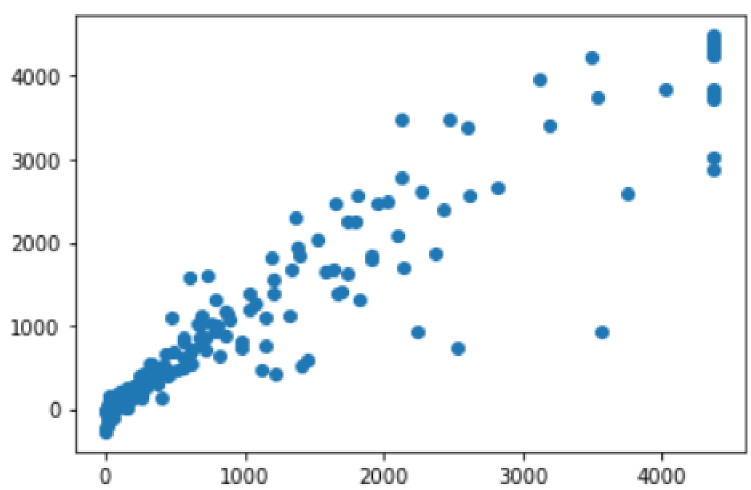
The condition number is large, 9.82e+03

This might indicate that there are strong multicollinearity or other numerical problem.

Mean Square Error(MSE)

```
402.4038730143032
```

```
<matplotlib.collections.PathCollection at 0x2dcaa30ab20>
```



Plot of predicted y VS Actual y

Observation

- From above fig we can see a linear trend in the dataset
- We see the variable sp500 is the most important or governing parameter.
- The model score for training and testing data is almost same, hence the model is good fitted model.
- The accuracy of the model is appproximately is **67.03% .**

Conclusion:

- From all the model we have seen above , we can say that
- Linear Regeression using sklearn ,statsmodel behave in similar way as they have same score.
- Hence we select the model of linear regression using statsmodel because it provides additional statistical information of the model

and the dataset which helps us to better understand the data and make better decisions overall.

Multicollinearity check using variation inflation factor

```
capital ---> 8.395261454766336
patents ---> 4.111535172959012
randd ---> 4.70591369415132
employment ---> 8.845053042338
sp500 ---> 3.742774549016143
tobinq ---> 1.8221996689301851
value ---> 8.97884183349674
institutions ---> 3.0031422135132573
```

Since most of the variables have VIF values greater than 1, hence multi-collinearity exists in the dataset.

Conclusion

Final linear regression equation for the target variable 'sales'

```
(28.81) * Intercept + (0.43) * capital + (-4.71) * patents + (0.59) * randd + (79.64) * employment + (169.71) * sp500 + (-16.4
8) * tobinq + (0.23) * value + (0.09) * institutions +
```

Here,

- The positive coefficient mean when the corresponding variables increases, sales will also increases
- The negative coefficient mean when the corresponding variable increases, sales will decreases.

Question 1.4

Inference:Based on these Predictions what are the business insights and recommendations

Answer:

Now since we have established a investing firm and working on 759 firms, we can find the best attributes to segregate higher and lower profitable in sales.

- The most important attribute in investing firm is sales of the ‘employment’ as it has highest positive coefficient
- As per the research a less sale as well as close to 50% correction in category such as patents and institutions but it is showing negative

correlation at tobniq which is we can predict a loss in particular category

- From all the model we have seen above , we can say that
- Linear Regeression using sklearn ,statsmodel behave in similar way as they have same score.
- As per the capital report we have enough capital available for spending which is close to 91% available.
- As we can analysis that 48% sale from the patents by the institutions which is helping for increasing the sale ratio.
- Sale is directly proportional to employment sales is increasing so number of employment also increasing
- Tobniq is also known as a q ratio and kaidors as its sale increasing so it will impact the physical assets in market and values will increase gradually.

Problem 2

You are hired by the Government to do an analysis of car crashes. You are provided details of car crashes, among which some people survived and some didn't. You have to help the government in predicting whether a person will survive or not on the basis of the information given in the data set so as to provide insights that will help the government to make stronger laws for car manufacturers to ensure safety measures. Also, find out the important factors on the basis of which you made your predictions.

Data Dictionary for Car_Crash

1. dvcat: factor with levels (estimated impact speeds) 1-9km/h, 10-24, 25-39, 40-54, 55+
2. weight: Observation weights, albeit of uncertain accuracy, designed to account for varying sampling probabilities. (The inverse probability weighting estimator can be used to demonstrate causality when the researcher cannot conduct a controlled experiment but has observed data to model)
3. Survived: factor with levels Survived or not_survived
4. airbag: a factor with levels none or airbag
5. seatbelt: a factor with levels none or belted
6. frontal: a numeric vector; 0 = non-frontal, 1=frontal impact
7. sex: a factor with levels f: Female or m: Male

8. ageOFocc: age of occupant in years
9. yearacc: year of accident
10. yearVeh: Year of model of vehicle; a numeric vector
11. abcat: Did one or more (driver or passenger) airbag(s) deploy? This factor has levels deploy, nodeploy and unavail
12. occRole: a factor with levels driver or pass: passenger
13. deploy: a numeric vector: 0 if an airbag was unavailable or did not deploy; 1 if one or more bags deployed.
14. injSeverity: a numeric vector; 0: none, 1: possible injury, 2: no incapacity, 3: incapacity, 4: killed; 5: unknown, 6: prior death
15. caseid: character, created by pasting together the populations sampling unit, the case number, and the vehicle number. Within each year, use this to uniquely identify the vehicle.

Importing Important Libraries

Question 2.1

Data ingestion: Read the dataset .do the descriptive statistics and do the null value condition check,write an inference on it.perform univariate and bivariate analysis .do exploratory data analysis

Answer:

	Unnamed: 0	dvcat	weight	Survived	airbag	seatbelt	frontal	sex	ageOFocc	yearacc	yearVeh	abcat	occRole	deploy	injSeverity	caseid
0	0	55+	27.078	Not_Survived	none	none	1	m	32	1997	1987	unavail	driver	0	4.0	02:13:02
1	1	25-39	89.627	Not_Survived	airbag	belted	0	f	54	1997	1994	nodeploy	driver	0	4.0	02:17:01
2	2	55+	27.078	Not_Survived	none	belted	1	m	67	1997	1992	unavail	driver	0	4.0	0.138206019
3	3	55+	27.078	Not_Survived	none	belted	1	f	64	1997	1992	unavail	pass	0	4.0	0.138206019
4	4	55+	13.374	Not_Survived	none	none	1	m	23	1997	1986	unavail	driver	0	4.0	04:58:01

Head of dataframe

	Unnamed: 0	dvcat	weight	Survived	airbag	seatbelt	frontal	sex	ageOFocc	yearacc	yearVeh	abcat	occRole	deploy	injSeverity	caseid
11212	11212	25-39	3179.688	survived	none	belted	1	m	17	2002	1985	unavail	driver	0	0.0	82:107:1
11213	11213	10-24	71.228	survived	airbag	belted	1	m	54	2002	2002	nodeploy	driver	0	2.0	82:108:2
11214	11214	10-24	10.474	survived	airbag	belted	1	f	27	2002	1990	deploy	driver	1	3.0	82:110:1
11215	11215	25-39	10.474	survived	airbag	belted	1	f	18	2002	1999	deploy	driver	1	0.0	82:110:2
11216	11216	25-39	10.474	survived	airbag	belted	1	m	17	2002	1999	deploy	pass	1	0.0	82:110:2

Tail of dataframe

Shape of dataframe

```
(11217, 16)
```

Info of dataframe

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 11217 entries, 0 to 11216
Data columns (total 16 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   Unnamed: 0   11217 non-null  int64  
 1   dvcat        11217 non-null  object 
 2   weight       11217 non-null  float64
 3   Survived     11217 non-null  object 
 4   airbag       11217 non-null  object 
 5   seatbelt     11217 non-null  object 
 6   frontal      11217 non-null  int64  
 7   sex          11217 non-null  object 
 8   ageOFocc     11217 non-null  int64  
 9   yearacc      11217 non-null  int64  
 10  yearVeh      11217 non-null  int64  
 11  abcat        11217 non-null  object 
 12  occRole      11217 non-null  object 
 13  deploy       11217 non-null  int64  
 14  injSeverity  11140 non-null  float64
 15  caseid       11217 non-null  object 
dtypes: float64(2), int64(6), object(8)
memory usage: 1.4+ MB
```

From above we can see that all variables having non-null values of 11217 and only one variable 'injseeverity' having null value 77.

Description of dataframe

	count	mean	std	min	25%	50%	75%	max
Unnamed: 0	11217.0	5608.000000	3238.213319	0.0	2804.000	5608.000	8412.000	11216.00
weight	11217.0	431.405309	1406.202941	0.0	28.292	82.195	324.056	31694.04
frontal	11217.0	0.644022	0.478830	0.0	0.000	1.000	1.000	1.00
ageOFocc	11217.0	37.427654	18.192429	16.0	22.000	33.000	48.000	97.00
yearacc	11217.0	2001.103236	1.056805	1997.0	2001.000	2001.000	2002.000	2002.00
yearVeh	11217.0	1994.177944	5.658704	1953.0	1991.000	1995.000	1999.000	2003.00
deploy	11217.0	0.389141	0.487577	0.0	0.000	0.000	1.000	1.00
injSeverity	11140.0	1.825583	1.378535	0.0	1.000	2.000	3.000	5.00

Duplicates and null values

```
new_df.duplicated().sum()
```

```
0
```

```
new_df.isnull().sum()
```

```
Unnamed: 0      0
dvcat           0
weight          0
Survived        0
airbag          0
seatbelt        0
frontal         0
sex             0
ageOFocc        0
yearacc         0
yearVeh         0
abcat           0
occRole         0
deploy          0
injSeverity    77
caseid          0
dtype: int64
```

As check we have an unwanted column in the dataframe 'unnamed: 0'

So we need to remove this column .

After removing this column - 'unnamed: 0' the data frame looks like

	dvcat	weight	Survived	airbag	seatbelt	frontal	sex	ageOFocc	yearacc	yearVeh	abcat	occRole	deploy	injSeverity	caseid
0	55+	27.078	Not_Survived	none	none	1	m	32	1997	1987	unavail	driver	0	4.0	02:13:02
1	25-39	89.627	Not_Survived	airbag	belted	0	f	54	1997	1994	nodeploy	driver	0	4.0	02:17:01
2	55+	27.078	Not_Survived	none	belted	1	m	67	1997	1992	unavail	driver	0	4.0	0.138206019
3	55+	27.078	Not_Survived	none	belted	1	f	64	1997	1992	unavail	pass	0	4.0	0.138206019
4	55+	13.374	Not_Survived	none	none	1	m	23	1997	1986	unavail	driver	0	4.0	04:58:01

Observation

- From above fig of info we can say that we are having 77 NaN values .
- So as per check in dataframe we found the NaN values in 77 different rows

	dvcat	weight	Survived	airbag	seatbelt	frontal	sex	ageOFocc	yearacc	yearVeh	abcat	occRole	deploy	injSeverity	caseid
767	10-24	10.579	survived	airbag	none	1	f	97	2000	2000	deploy	driver	1	NaN	49:78:1
1025	40-54	9.780	survived	none	none	1	f	97	2000	1984	unavail	driver	0	NaN	72:65:1
1121	40-54	23.081	survived	airbag	none	0	f	68	2000	1999	nodeploy	driver	0	NaN	73:44:01
1576	10-24	52.779	survived	airbag	belted	1	m	25	2000	1999	nodeploy	pass	0	NaN	75:02:02
1594	25-39	56.280	survived	airbag	belted	0	f	53	2000	1999	nodeploy	pass	0	NaN	75:09:02
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
10460	10-24	548.186	survived	airbag	belted	1	m	25	2002	1999	deploy	pass	1	NaN	75:47:01
10464	10-24	440.426	survived	airbag	belted	1	m	22	2002	1999	nodeploy	pass	0	NaN	75:49:01
10980	10-24	20.409	survived	none	belted	1	f	17	2002	1987	unavail	pass	0	NaN	81:10:02
11042	25-39	392.914	survived	airbag	belted	1	f	59	2002	1999	nodeploy	pass	0	NaN	81:71:1
11169	10-24	569.739	survived	none	belted	1	m	22	2002	1990	unavail	pass	0	NaN	82:59:01

77 rows × 15 columns

From above data frame as we have made some changes

We got the value

1 – 9km/h because of this value we were getting an error while predicting the value.

So we changed the value from 1 – 9km/h to 1 – 9 .

As we have check value counts from all the variables .

```
df.dvcat.value_counts()
```

```
10-24    5414
25-39    3368
40-54    1344
55+       809
1-9       282
Name: dvcat, dtype: int64
```

```
df.Survived.value_counts()
```

```
survived        10037
Not_Survived     1180
Name: Survived, dtype: int64
```

```
df.airbag.value_counts()
```

```
airbag    7064
none      4153
Name: airbag, dtype: int64
```

```
df.seatbelt.value_counts()
```

```
belted    7849
none      3368
Name: seatbelt, dtype: int64
```

```
df.sex.value_counts()
```

```
m    6048
f    5169
Name: sex, dtype: int64
```

```
df.abcat.value_counts()
```

```
deploy      4365
```

Observations :

- From above observations we knew that we have 14 different variable named
- {dvcat , weight , Survived , airbag , seatbelt , frontal , sex , ageofocc , yearacc , yearVeh , abcat , occRole , deploy , injseverity , caseid}
- The above dataset contains 11217 rows and 16 columns.
- The variables dvcat,survived,seatbelt,airbag,sex,abcat,occrole,caseid are of object datatype, and rest variable are integer and float datatype.
- There are no null & duplicates values in dataset
- The summary table shows mean,median,standard deviation, minimum and maximum value,etc for all the variables.

Univariate Analysis

1)outlier Identification

<AxesSubplot:>

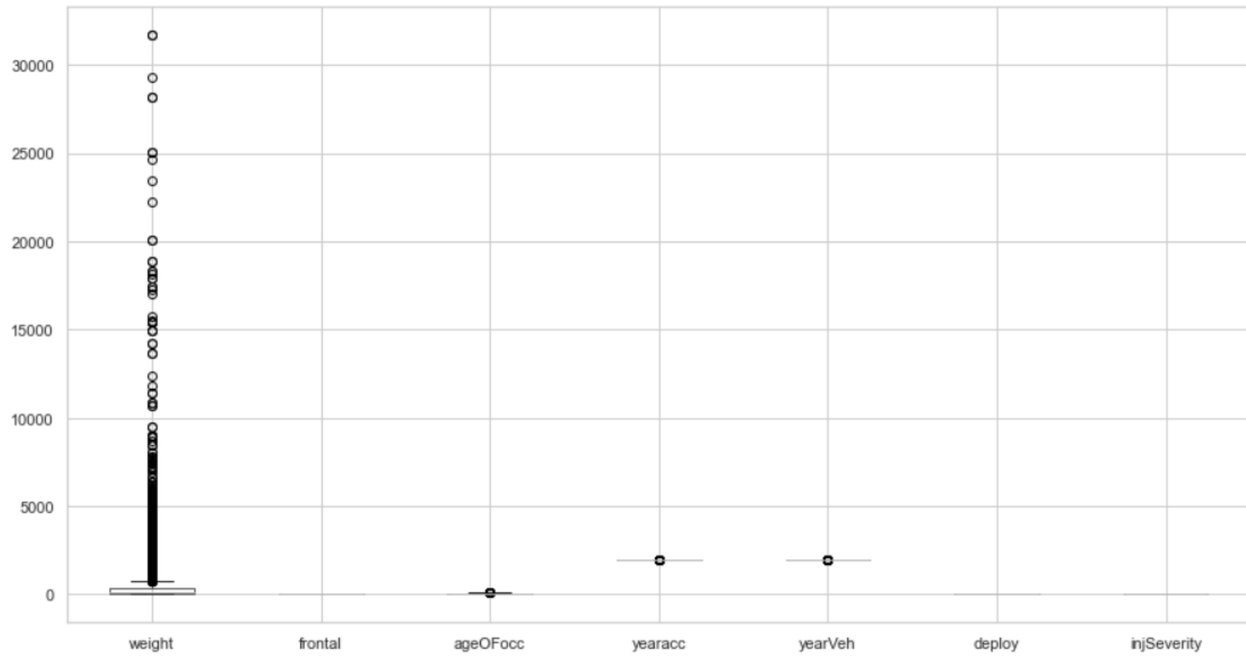


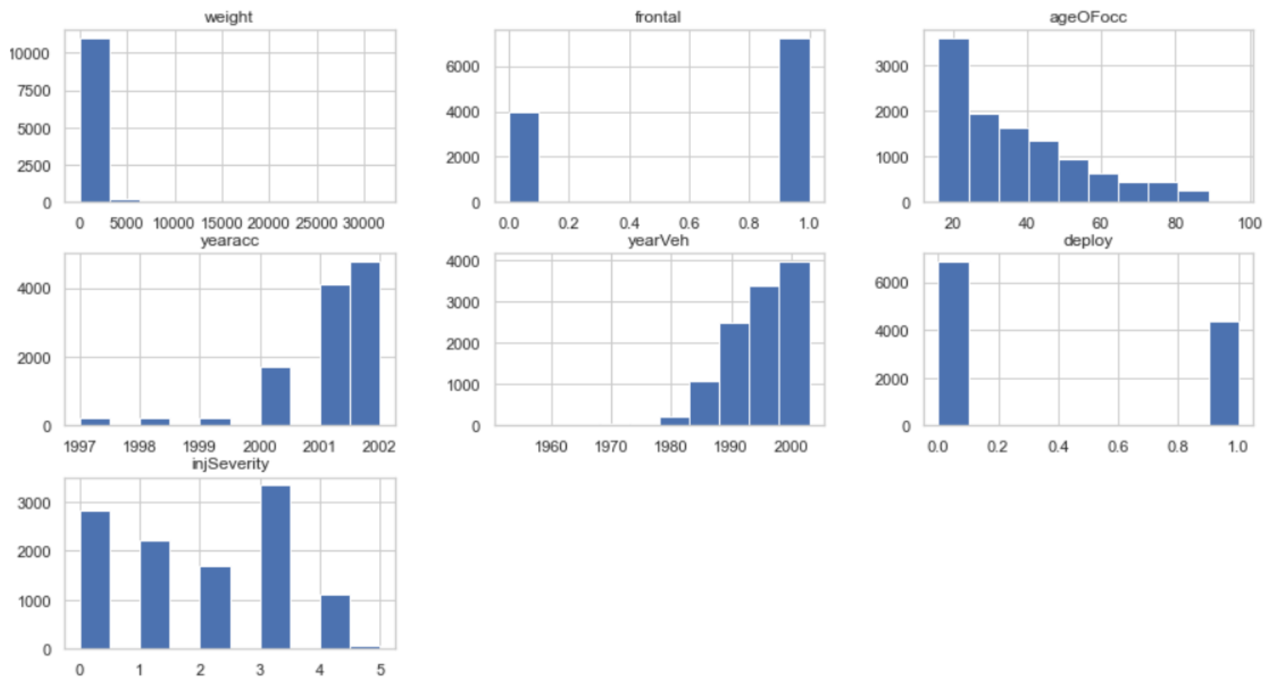
Figure : boxplot for outlier identification

Observation :

From the plot above , we can say that

- **The variable weight contains a good number of outliers.**
- **The variable ageOFocc , yearacc , yearVeh has very dew outlier.**
- **The variable frontal , deploy , injseverity contains no outliers.**
- **Treating outliers sometimes results in the models having better performance but the models loose out on generalization. Hence , we be treating them in order to not loose out on generalization.**

Distribution check



Histplot for distribution check for all variables

```
weight          11.115386
frontal         -0.601667
ageOFocc         0.911059
yearacc         -1.671687
yearVeh         -1.026743
deploy           0.454813
injSeverity      0.046053
dtype: float64
```

Observation:

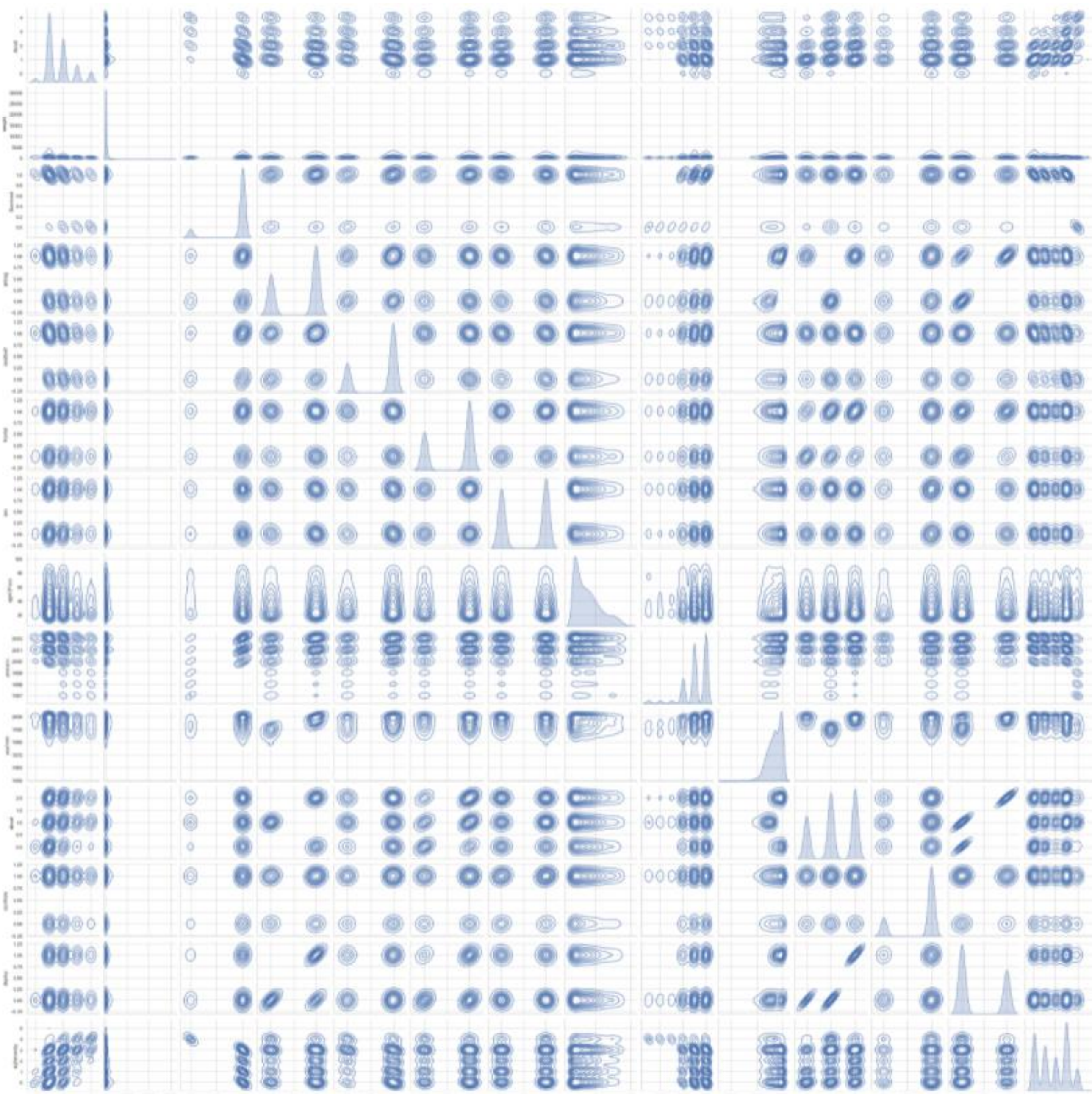
From the above graph & skew values above , we can say that

- Since the skewness value of variables ‘frontal’ , ‘deploy’ , ‘injseverity’ is between -0.5 and +0.5 they show approximately symmetric distribution.
- Since the skewness value of variables ‘ yearacc’ , ‘yearVeh’ is between -2 and -0.5
- Since the skewness value of variable ‘weight ‘ is greater that +1 ,it shows highly skewed distribution

Multivariate Analysis

1)correlation check

	weight	frontal	ageOFocc	yearacc	yearVeh	deploy	injSeverity
weight	1.000000	0.000659	-0.040111	0.056892	-0.015226	-0.065783	-0.220400
frontal	0.000659	1.000000	-0.048856	0.059768	-0.024267	0.260388	-0.054496
ageOFocc	-0.040111	-0.048856	1.000000	-0.072271	-0.002070	-0.009556	0.124169
yearacc	0.056892	0.059768	-0.072271	1.000000	0.247743	0.091252	-0.303666
yearVeh	-0.015226	-0.024267	-0.002070	0.247743	1.000000	0.452448	-0.140054
deploy	-0.065783	0.260388	-0.009556	0.091252	0.452448	1.000000	0.038374
injSeverity	-0.220400	-0.054496	0.124169	-0.303666	-0.140054	0.038374	1.000000



Observation:

From the graph above , we can say that

- There is a weak correlation between the variables.
- (weight and frontal)

- The good correlation between variables are very few but they are strongly bond.as in
- Dvcat – injseverity (0.47)
- Survived – yearacc (0.55)
- Yearveah - airbag (0.77)
- Deploy – airbag (0.61)
- From the above observation we can say that the passenger with seatbelt having less chances of dying when the car crashes.
- Or the passanger without seatbelt having more chances of dying when car crashes.
- 1st the Safety measures should be taken by the passenger.
- As every vehicle should have air bag for every passenger so that even the car met with an accident the passenger inside the car should be safe .

Question 2.2

Encode the data (having string values) for Modelling.
Data Split: Split the data into train and test

(70:30).Apply Logistic Regression and LDA (linear discriminant analysis).

Encoding object datatype

Before imputing the datatype

```
dvcat          object
weight        float64
Survived       object
airbag         object
seatbelt       object
frontal         int64
sex            object
ageOFocc        int64
yearacc         int64
yearVeh         int64
abcat          object
occRole        object
deploy          int64
injSeverity   float64
caseid         object
dtype: object
```

After imputing the datatype

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 11217 entries, 0 to 11216
Data columns (total 14 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   dvcat        11217 non-null  int64
 1   weight       11217 non-null  float64
 2   Survived     11217 non-null  float64
 3   airbag       11217 non-null  float64
 4   seatbelt     11217 non-null  float64
 5   frontal      11217 non-null  int64
 6   sex          11217 non-null  float64
 7   ageOFocc     11217 non-null  int64
 8   yearacc      11217 non-null  int64
 9   yearVeh      11217 non-null  int64
 10  abcat        11217 non-null  float64
 11  occRole      11217 non-null  float64
 12  deploy       11217 non-null  int64
 13  injSeverity  11217 non-null  float64
dtypes: float64(8), int64(6)
memory usage: 1.2 MB
```

Once , encoding is done we further,

1)copy all the predictor variables into a dataframe and copy target into another dataframe

2)we split the entire dataset into training and testing (70:30) ratio respectively. We do this to make sure models learn considering good amount of data .

3)then we build and fit the data into the models namely a)Logistic Regression

b)Linear Discriminent analysis

4)Once the model is built we then predict the same for both training and testing .we do this to understand how the model behaves with the new data.

5)Then , we predict the classes and probabilities which helps us later to evaluate the model performance.

After droping the unwanted column Unnamed and caseid

	dvcat	weight	Survived	airbag	seatbelt	frontal	sex	ageOFocc	yearacc	yearVeh	abcat	occRole	deploy	injSeverity
0	55+	27.078	Not_Survived	none	none	1	m	32	1997	1987	unavail	driver	0	4.0
1	25-39	89.627	Not_Survived	airbag	belted	0	f	54	1997	1994	nodeploy	driver	0	4.0
2	55+	27.078	Not_Survived	none	belted	1	m	67	1997	1992	unavail	driver	0	4.0
3	55+	27.078	Not_Survived	none	belted	1	f	64	1997	1992	unavail	pass	0	4.0
4	55+	13.374	Not_Survived	none	none	1	m	23	1997	1986	unavail	driver	0	4.0

Applying grid search method

Y-test predict probability

	0	1
0	9.530469e-01	0.046953
1	9.802788e-01	0.019721
2	4.772795e-10	1.000000
3	5.260633e-07	0.999999
4	5.929434e-02	0.940706

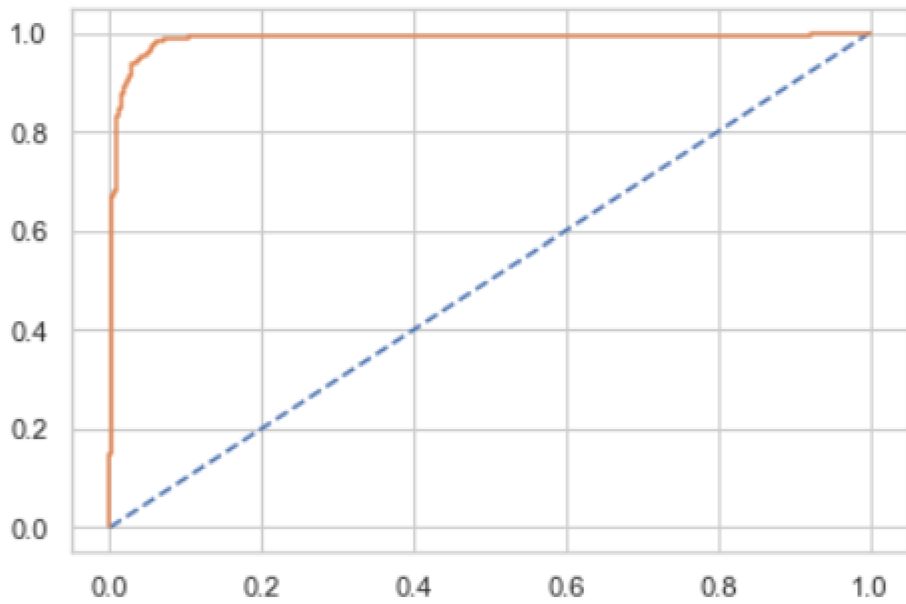
Y-train predict probability

	0	1
0	1.280884e-02	0.987191
1	2.049655e-02	0.979503
2	5.028476e-04	0.999497
3	4.165189e-06	0.999996
4	4.892341e-09	1.000000

AUC and ROC for training data

```
AUC: 0.987
```

```
[<matplotlib.lines.Line2D at 0x196384bf160>]
```

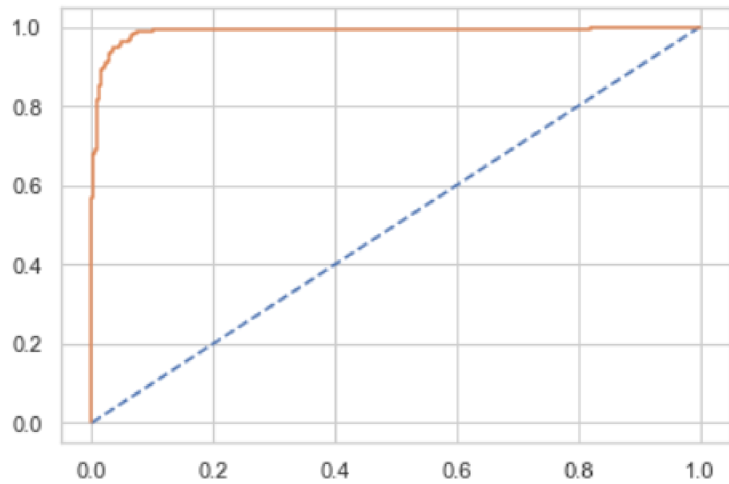


Best model score for training `0.9812762705387849`

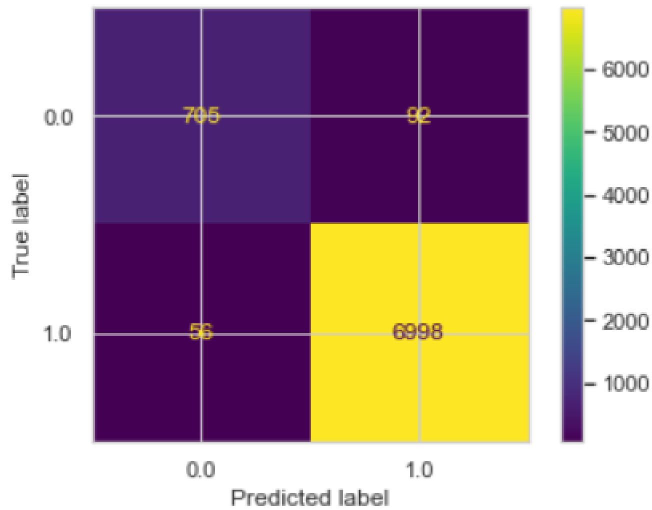
AUC and ROC curve for testing data

```
AUC: 0.988
```

```
[<matplotlib.lines.Line2D at 0x1963a513a60>]
```



Confusion matrix and classification report on training data



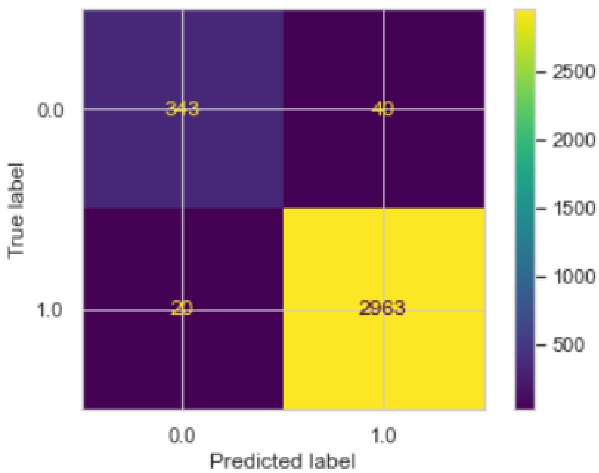

```
print(classification_report(y_train, ytrain_predict))
```

```
              precision    recall  f1-score   support

         0.0       0.93      0.88      0.91       797
         1.0       0.99      0.99      0.99      7054

    accuracy                           0.98      7851
   macro avg       0.96      0.94      0.95      7851
weighted avg       0.98      0.98      0.98      7851
```

Confusion matrix and classification report for testing data




```python
print('Confusion Matrix','\n',metrics.confusion_matrix(y_test, ytest_predict),'\n')
print('Classification Report','\n',metrics.classification_report(y_test, ytest_predict))
```

```
Confusion Matrix
 [[ 343   40]
 [  20 2963]]

Classification Report
               precision    recall  f1-score   support

         0.0       0.94      0.90      0.92       383
         1.0       0.99      0.99      0.99      2983

    accuracy                           0.98      3366
   macro avg       0.97      0.94      0.95      3366
weighted avg       0.98      0.98      0.98      3366
```

Logistic Regression Logistic regression is a linear model for classification rather than regression. It is also known as logit regression. In this model, the probabilities describing the possible outcomes of a single trial are modeled using a logistic function.

Note: - Regularization is applied by default, which is common in machine learning but not in statistics.

- Another advantage of regularization is that it improves numerical stability. No regularization amounts to setting C to a very high value.

There are two methods to solve a Logistic Regression problem:

1. Stats Model
2. Scikit Learn

Here, we will use Grid Search (scikit learn method) to find the optimal hyperparameters of a model which results in the most 'accurate' predictions and get the best parameters.

The parameters used in GridsearchCV can be explained as :

- param_grid : requires a list of parameters and the range of values for each parameter of the specified estimator
- estimator: requires the model we are using for the hyper parameter tuning process
- cross-validation (cv):performed in order to determine the hyper parameter value set which provides the best accuracy levels.
- N_jobs: controls the number of cores on which the package will attempt to run in parallel.

• solver is a string ('liblinear' by default) that decides what solver to use for fitting the model. Other options are'newton-cg', 'lbfgs', 'sag', and 'saga'.

• max_iter is an integer (100 by default) that defines the maximum number of iterations by the solver during model fitting.

• verbose is a non-negative integer (0 by default) that defines the verbosity for the 'liblinear' and 'lbfgs' solvers.

Using the confusion matrix, the True Positive, False Positive, False Negative, and True Negative values can be extracted which will aid in the calculation of the accuracy score, precision score, recall score, and f1score.

Question 2.3

Performance Metrics: Check the performance of Predictions on Train and Test sets using Accuracy, Confusion Matrix, Plot ROC curve and get ROC_AUC

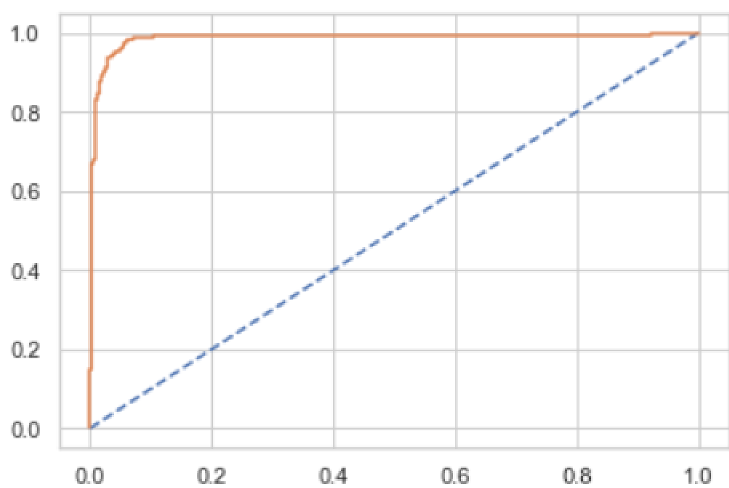
score for each model.Compare both the models and write inferences, which model is best/optimized.

Once we have build the model and predicted the values we check its performance matrix

1) Model accuracy for training and test data

AUC – 0.987

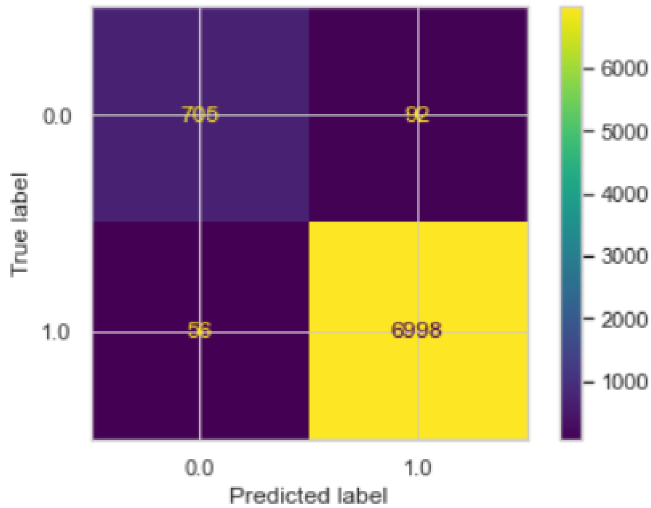
```
[&lt;matplotlib.lines.Line2D at 0x196384bf160&gt;]
```



2) Confusion matrix and classification report for training data

```
array([[ 705,   92],
       [  56, 6998]], dtype=int64)
```

```
plot_confusion_matrix(LogR,X_train,y_train);
```




```
print(classification_report(y_train, ytrain_predict))
```

```
              precision    recall  f1-score   support

         0.0       0.93      0.88      0.91       797
         1.0       0.99      0.99      0.99      7054

    accuracy                           0.98      7851
   macro avg       0.96      0.94      0.95      7851
weighted avg       0.98      0.98      0.98      7851
```

Training data

True positive:705

False Positive:56

False Negative:92

True Negative:6998

AUC: 0.987%

Accuracy: 0.98%

Precision: 0.99%

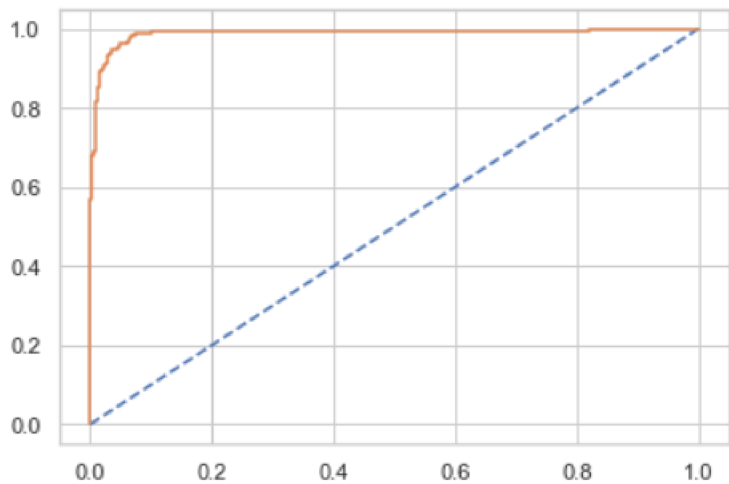
f1-Score: 0.99%

Recall:0.99%

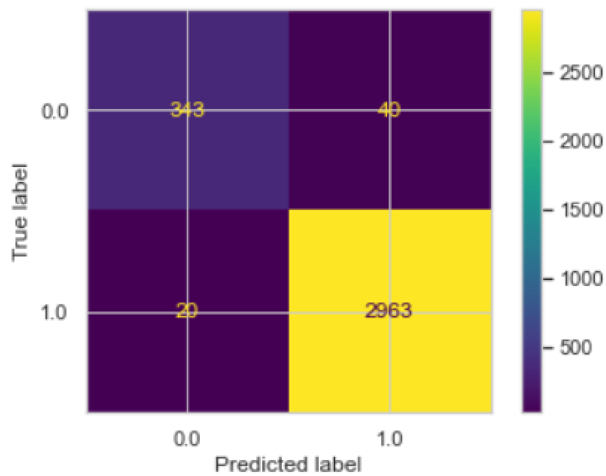
Model accuracy for testing data

AUC-0.988

```
[<matplotlib.lines.Line2D at 0x1963a513a60>]
```



3) Confusion matrix and classification report for testing data




```
print('Confusion Matrix','\n',metrics.confusion_matrix(y_test, ytest_predict),'\n')
print('Classification Report','\n',metrics.classification_report(y_test, ytest_predict))
```

```
Confusion Matrix
 [[ 343   40]
 [  20 2963]]

Classification Report
               precision    recall  f1-score   support

         0.0       0.94      0.90      0.92       383
         1.0       0.99      0.99      0.99      2983

    accuracy                           0.98      3366
   macro avg       0.97      0.94      0.95      3366
weighted avg       0.98      0.98      0.98      3366
```

Testing data

True positive:343

False Positive:20

False Negative:40

True Negative:2963

AUC: 0.988%

Accuracy: 0.98%

Precision: 0.99%

F1 score: 0.99%

Precision: 0.99%

Question 2.4

Inference: Based on these predictions, what are the insights and recommendations.

Answer:

From the analysis above we came to know that the linear Discriminant analysis performs much better

- We had a government problem where we need to predict whether how many people will survive car crash and how many not ,for this problem we had done both logistic regression and linear discriminant analysis .
- since both results are same but as compared to that we get much better result in LDA.
- As government want to do analysis on car crashed among the different aspects are dependent which is mentioned below such as if car crashed impact passenger is not deployed with seatbelt its less chances of survival.
- Car crashed frontal and airbag deployed as per analysis survival chance is 50%
- The majority of age is in between 20 to 40 years old and as we can observed major year of the accidents is 2000 to 2002
- Where few of them are 50% deployed so they survived