Machine Learning (CS60050)- Autumn 2020
Assignment 2

Submitted by:
Group No . 083
Ravi Pratap Singh -20CS60R60
Sahil Jain -20CS60R64

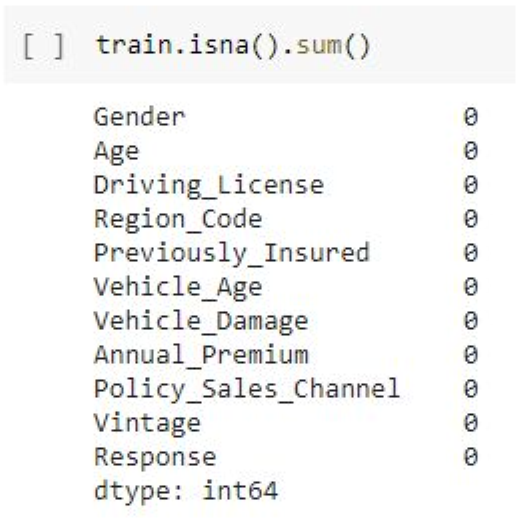
Task 1 :
Using Naive Bayes Classifier

Step 1 : Imported library *pandas* for data manipulation and *numpy* for numerical analysis and finding mean, variance etc.
Seaborn and matplotlib for data visualization and analysis (although that part is commented) .

Step 2 : Imported the dataset
Removed the id and driving licence column because it has no significant effect on our accuracy .
Randomly split data into 80-20 ratio for testing and training respectively .

PART(A)

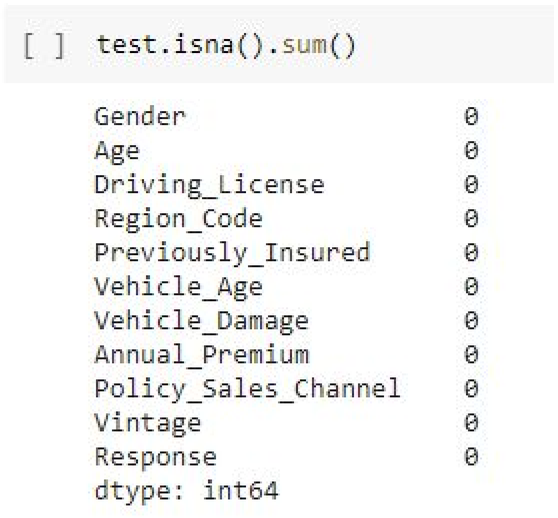
Step 3 : : *insa*() function is used to detect missing values. It returns a boolean same-sized object indicating if the values are NA. NA values, such as None or numpy.NaN, gets mapped to True values . Everything else gets mapped to False values .
So used this to find any missing value in the data set for both training and test dataset .
There were no missing values in the dataset.



```
[ ]  train.isna().sum()

     Gender                  0
     Age                     0
     Driving_License         0
     Region_Code             0
     Previously_Insured      0
     Vehicle_Age             0
     Vehicle_Damage          0
     Annual_Premium          0
     Policy_Sales_Channel    0
     Vintage                 0
     Response                0
     dtype: int64
```

No missing values in training data



```
[ ]  test.isna().sum()

     Gender                  0
     Age                     0
     Driving_License         0
     Region_Code             0
     Previously_Insured      0
     Vehicle_Age             0
     Vehicle_Damage          0
     Annual_Premium          0
     Policy_Sales_Channel    0
     Vintage                 0
     Response                0
     dtype: int64
```

No missing values in testing data

PART(B)

Step 4 : Encoded categorical variables such as Gender , Previously insured , Vehicle age , Vehicle damage using *sklearn.preprocessing* . Also converted the float type columns such as Annual Premium , Policy sales channel to integer to get data consistency among columns .
Above step is performed for both training and test data set.

Step 5: A class definition for feature scaling (Normalization) to normalise the data within a particular range. Functions to transform and fit the data is defined in the class .

Step 6: A class definition for Naive Bayes classifier which includes the function to fit the training data , a function for testing the data . Used the mean and standard deviation from the numpy .

Step 7: Separated train and test data set into X and Y dimension i.e separating target attribute from rest of the column .

PART(C)

Step 8: :Used the 5 fold cross validation on the training data set for hyper-parameter tuning and called the Naive bayes *‘ fit ‘* function to train our model . With each different alpha value dataset we train a model. Using cross-validation data we find the accuracy of our model. We consider that alpha value as right alpha which gives high accuracy.

Now tested our model with the actual ‘test’ dataset to get the final accuracy .Used the predict function of naive bayes class to get all predicted values, then found out the True positive , True negative , False positive ,False Negative to build the “Confusion Matrix “.
From the confusion matrix calculated the final accuracy as (tn+tp)/(tn+fp+fn+tp) that came around **86%**.

Sample confusion matrix :

```
[[66851    76]
 [ 9277    18]]
```

Task 2 :

PCA on processed data from part 1.

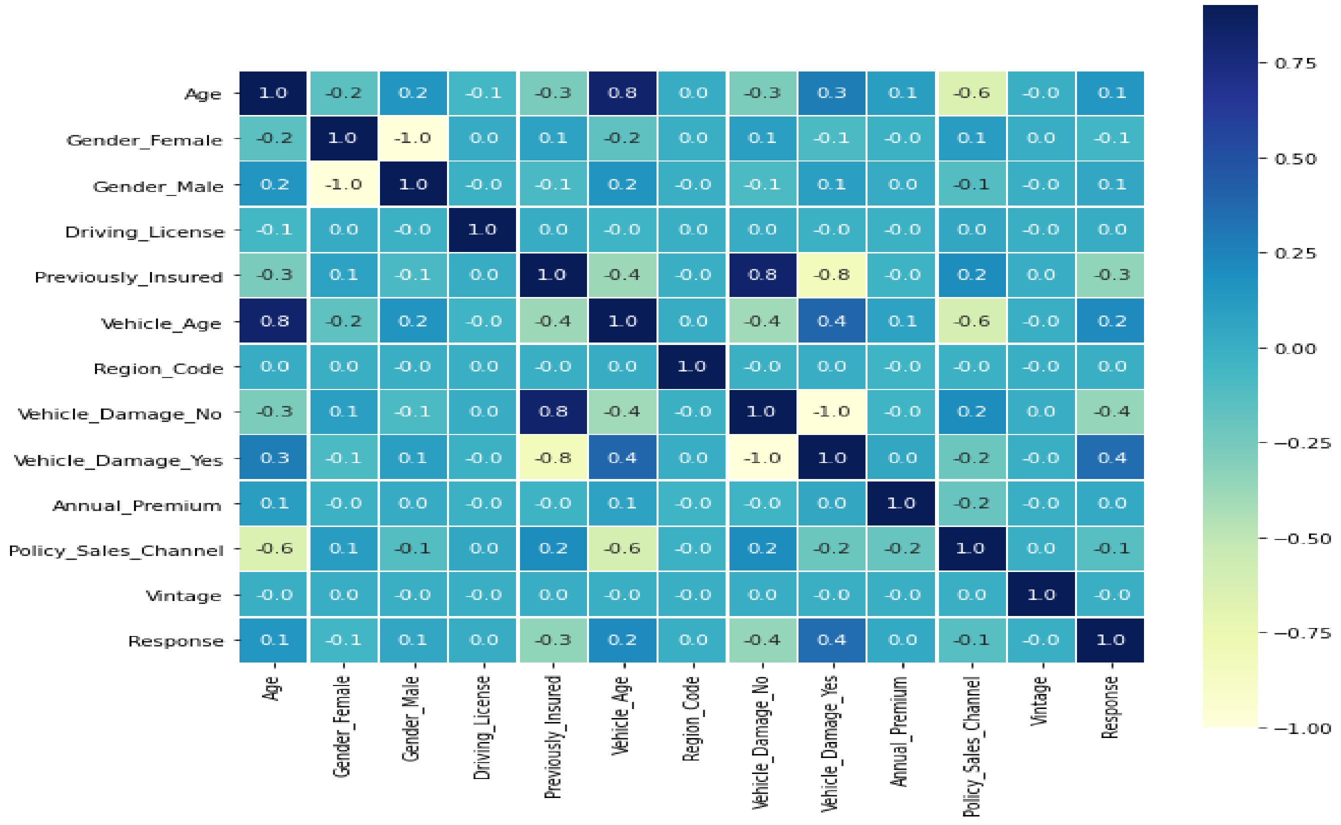
Step 1:PCA is affected by scale so we need to scale the features in our data before applying for a PCA. Using StandardScaler helped to standardize the data set's features onto unit scale.

PART (A)

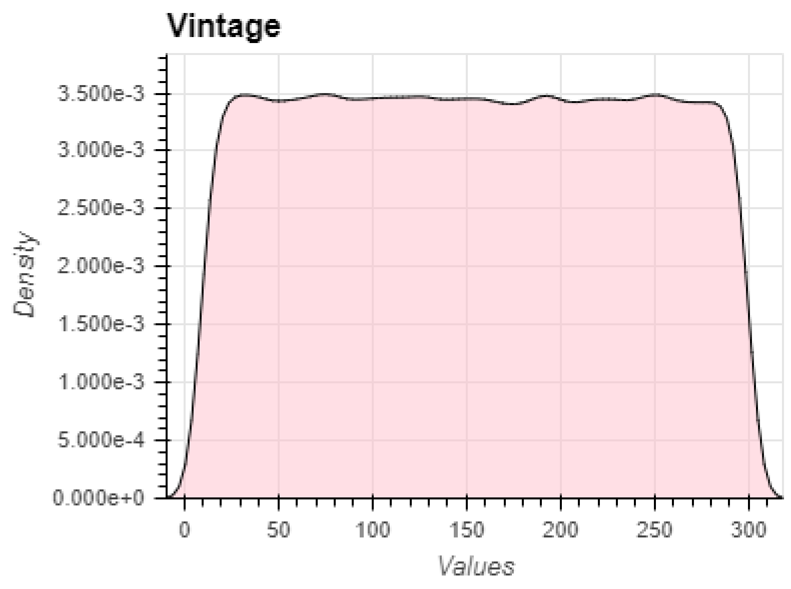
Step 2 : By creating the correlation matrix(measuring statistical dependence between the rankings of two variables). we got some correlation between Response and "Vehicle_Damage", furthermore there is a negative relation with the binary "Previously_Insured" variable. Another interesting relations:

- age is strongly correlated with age of car. This is well-known in actuarial world relation that young people drive old cars
- age with sales channel, old people tend to use brokers and agents, young ones use internet
- previously insured is correlated with age and vehicle age. Young people tend to change insurer often

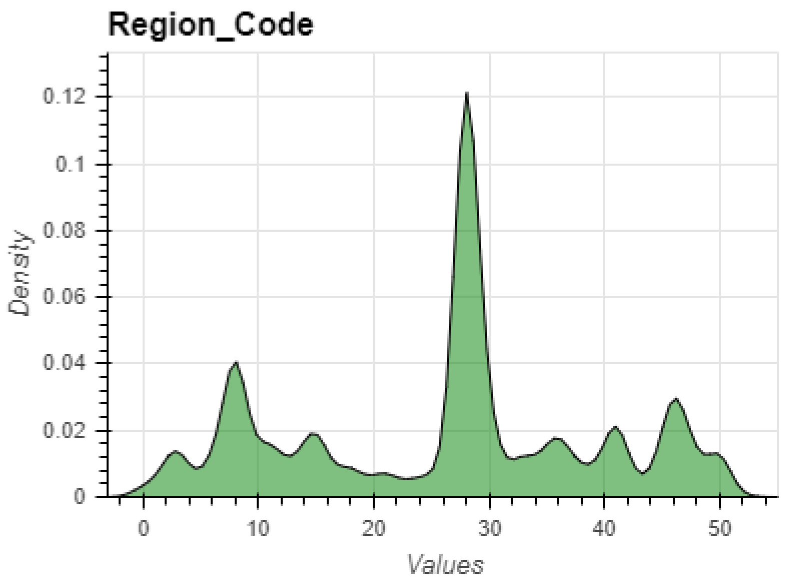
- previously insured with vehicle damage, indeed a lot of policyholders change the insurer while they correspond to big risk
- vehicle age and policy sales channel - this is an interesting spurious effect. The real underlying effect is relation between age and sales channel, indirectly influencing numbers for age of vehicl

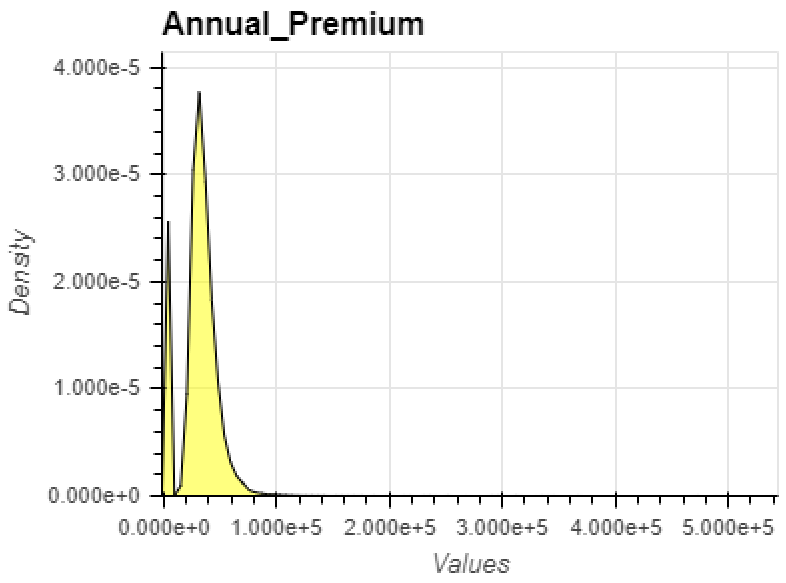


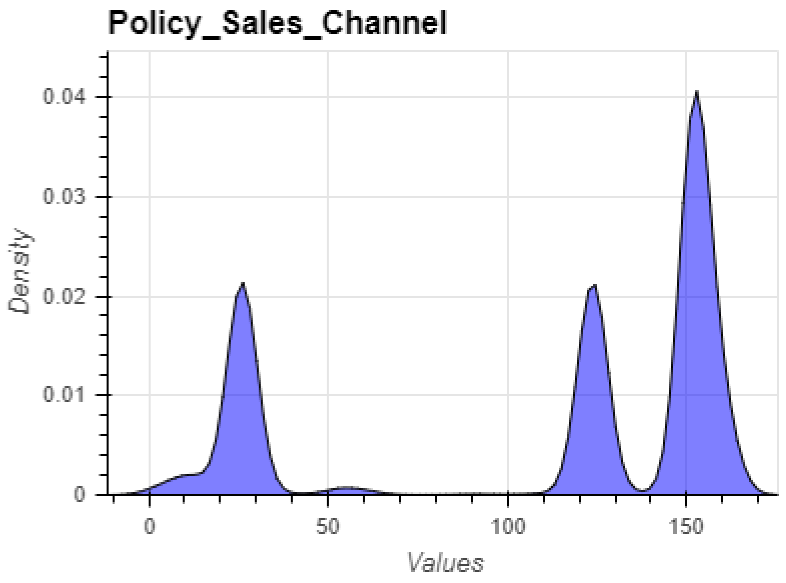
Step 3 : Factors distributions

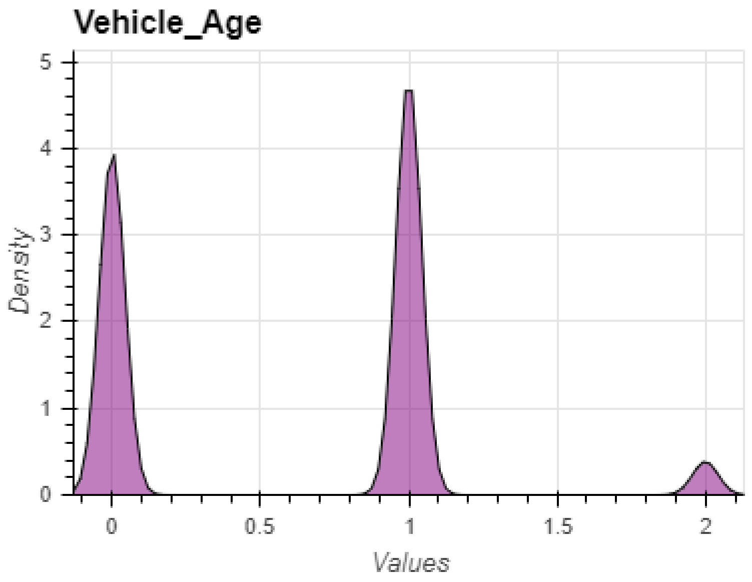


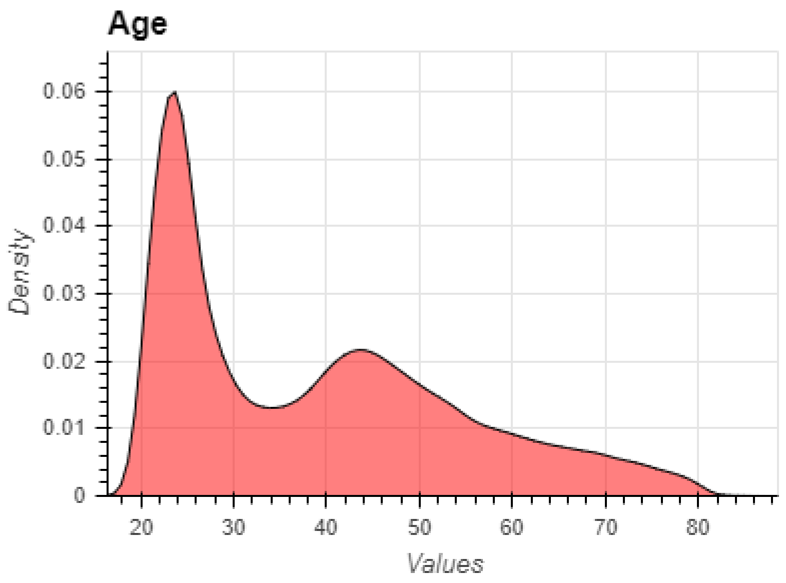
``


Region_Code
Density
Values
0
0.02
0.04
0.06
0.08
0.1
0.12
0
10
20
30
40
50


Annual_Premium
Density
Values
0.000e+0
1.000e-5
2.000e-5
3.000e-5
4.000e-5
0.000e+0
1.000e+5
2.000e+5
3.000e+5
4.000e+5
5.000e+5

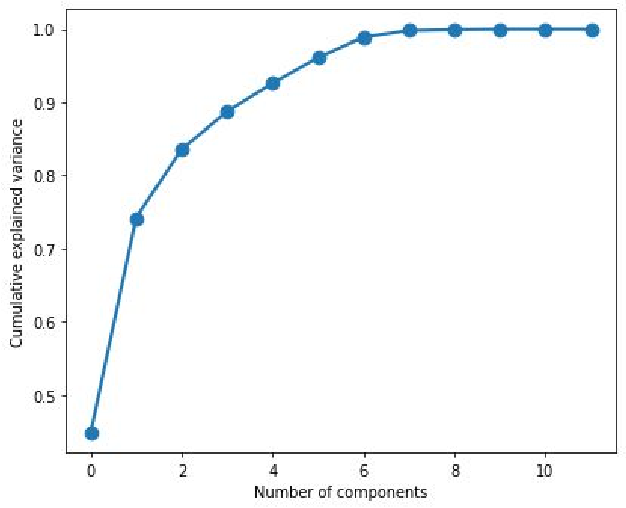

Policy_Sales_Channel
Density
Values
0
0.01
0.02
0.03
0.04
0
50
100
150

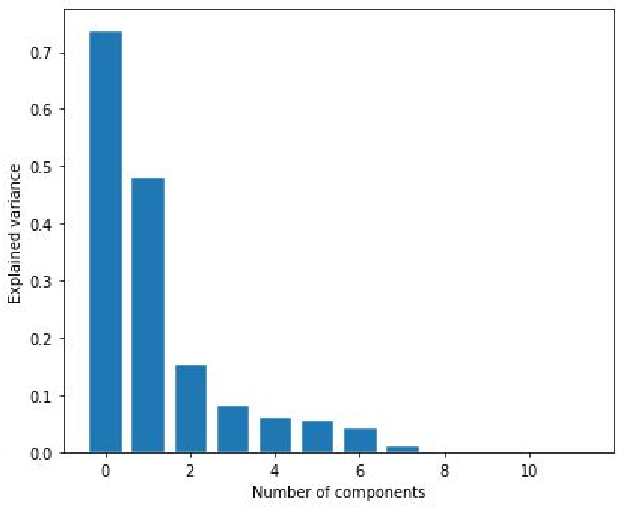




- Age has no regular shape. It is highly skewed towards younger ages .
- Region code, sales channel and vehicle age are dispersed around some dominating points.
- Annual premium has reasonable shape.
- Vintage is uniformly distributed, hence probably useless.

Step 4: We had applied a linear method: PCA which allows for assessment how many components suffice to cover the majority of variance(95%) in the data set. For this we look at cumulative variance .

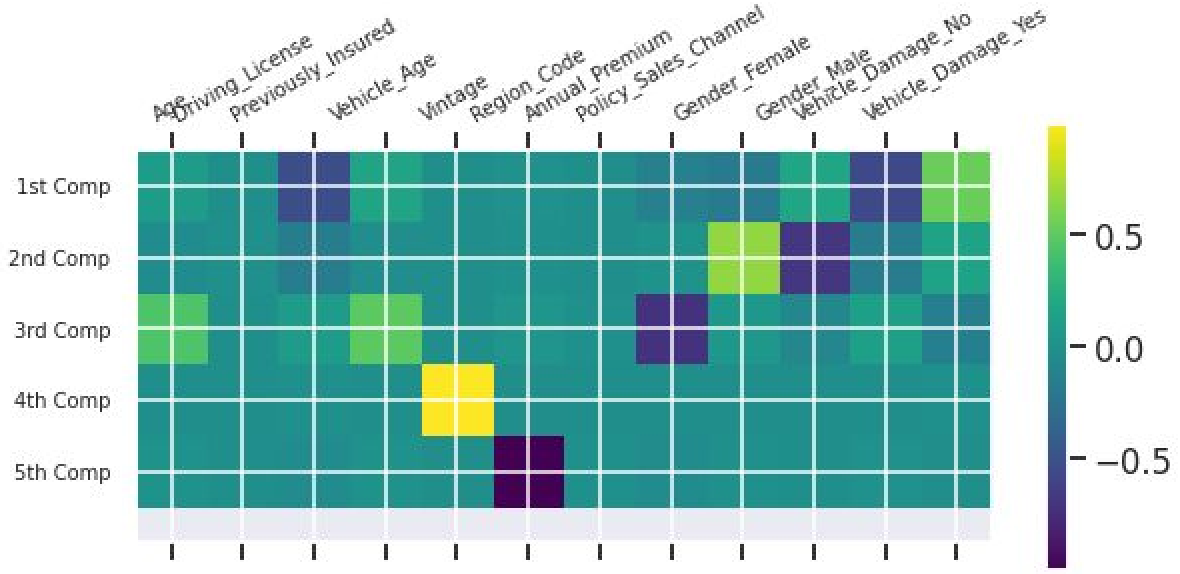




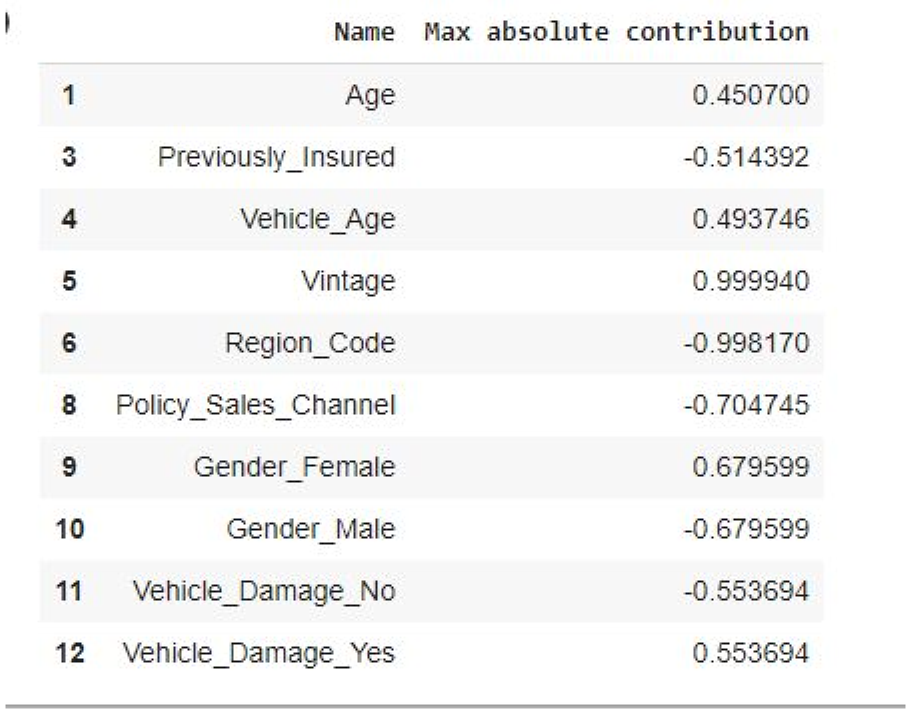
Step 5: The inbuilt function in *sklearn.decomposition* allows us to detect how many components are needed to capture the variability at a satisfying level. The A shortcoming of PCA is linearity assumption .

We got 5 components that will cover 95% of the variance .

Step 6: We applied regular PCA with 5 components and check what variables contribute the most:



In more readable format it will be as follows :



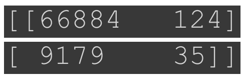
	Name	Max absolute contribution
1	Age	0.450700
3	Previously_Insured	-0.514392
4	Vehicle_Age	0.493746
5	Vintage	0.999940
6	Region_Code	-0.998170
8	Policy_Sales_Channel	-0.704745
9	Gender_Female	0.679599
10	Gender_Male	-0.679599
11	Vehicle_Damage_No	-0.553694
12	Vehicle_Damage_Yes	0.553694

To sum up:

- 'age' contributes to at least two components
- 'previously insured' contributes to at least three components
- 'vehicle age' performs same as 'age'
- 'vintage' dominates one and only one component 4th
- 'sales channel' contributes to at least two components
- both 'gender' and 'vehicle damage contribute to three components
- ‘Annual premium’ contributed to at least one component.
- ‘Policy sales channel’ contributed to zero components.

PART(B)

Step 7: Used the features extracted from PCA to train your model using naive bayes classifier and computed 5-fold cross validation on the training set.With the test data set(after applying PCA) calculated final accuracy and got it around **88%.**
Sample Confusion matrix from one of the runs



```
[[66884   124]
[ 9179    35]]
```

Task 3:

PART A: Outlier detection -

Using Z-score -

Computed the mean and standard deviation of the column using numpy.

Setting a threshold value of 3 as required in the question.

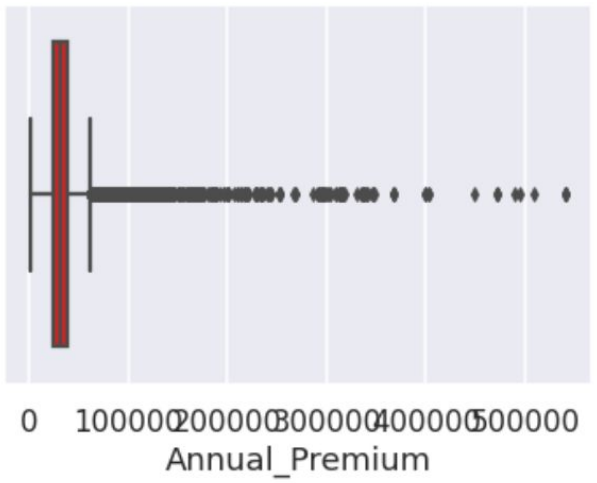
Formula for calculating Z-score = $(x - mean) / standard\ deviation$.

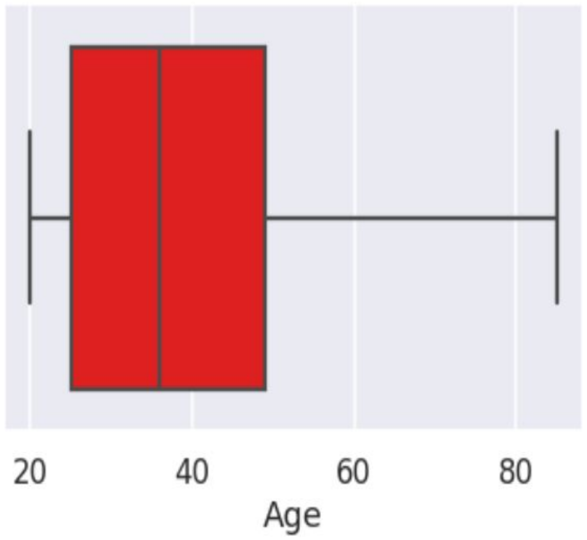
Step 1 - Created a function outlier_detection which checks whether sample data is outlier or not. If it is an outlier, add it to the list $outliers[]$.

Step 2- Used the above defined function on the features $Annual\ Premium$ and Age.

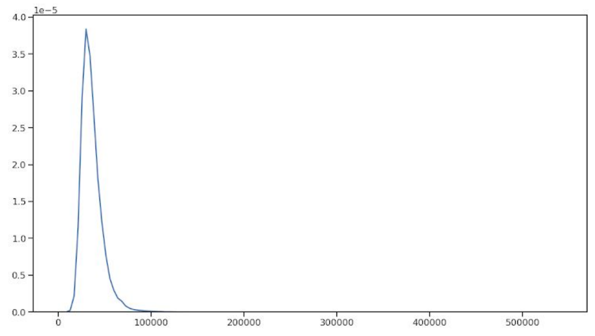
Step 3- Store the outliers of the above mentioned features in list data structures.

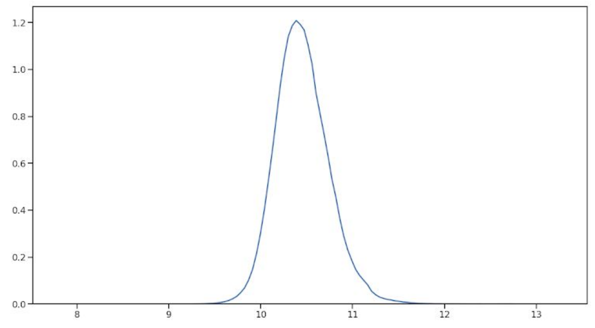
Step 4- Dropping the samples having maximum outliers.





Annual premium over logarithm range :





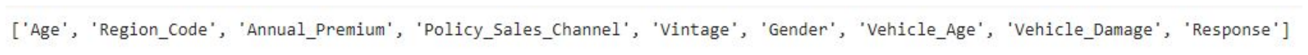
PART B: Using the sequential backward selection method-

We started with the original set of features and removed features one by one and checked the accuracy of the new training set using naive bayes model from task 1.

Also,stored the feature names which were removed to give better results in the list *feature_removed* [ ].

PART C : Print the final set of features formed-

Printed the list of final features.



```
['Age', 'Region_Code', 'Annual_Premium', 'Policy_Sales_Channel', 'Vintage', 'Gender', 'Vehicle_Age', 'Vehicle_Damage', 'Response']
```

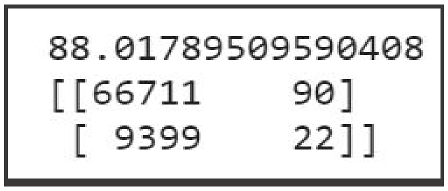
Compute 5-fold cross validation on the training set

Used the 5 fold cross validation on the training data set for hyper-parameter tuning and called the Naive bayes *‘ fit ‘* function to train our model . With each different alpha value dataset we train a model

PART D :

Printing the Final Results-

- Printed the final accuracy after applying all the tasks and the confusion matrix.



```
88.01789509590408
[[66711    90]
 [ 9399    22]]
```