LIFE INSURANCE SALES - CAPSTONE BUSINESS REPORT NOTES -2

THAKUR ARUN SINGH

JANUARY 2022

This Business Report shall provide detailed explanation of how we approached each problem given in the assignment. It shall also provide relative resolution and explanation with regards to the problems

CONTENTS

Problem 1: Model building and interpretation

The dataset belongs to a leading life insurance company. The company wants to predict the bonus for its agents so that it may design appropriate engagement activity for their high performing agents and up skill programs for low performing agents

PROBLEM 1.A

Build various models (You can choose to build models for either or all of descriptive, predictive or prescriptive purposes)

Resolution:

Project Approach

The work that we have completed:

We have created multiple models and applied them on different sets of data as required. All the different models which were created were then evaluated using the AUC / F1 score at the end for the testing data set. Based on this an optimal model was chosen. Eventually we also found the feature importance for the most optimal model.

Models Built

Various tree based as well as distance based models were built as part of this exercise using the different data sets as elaborated earlier. These models were built using sklearn and statsmodel libraries. There were various constraints, biggest one being the Type 2 error, which we had to minimize, as bonus variable was the main objective of this exercise. This will discuss in detail later in the report.

Model Tuning method

Various model tuning approaches were followed. Primarily we made use of GridSearchCV function with cv = 3 for model hyper parameter tuning. Also we had to tweak the threshold values to maximize the recall values. Threshold tweaking was required as we had a typical problem of recall precision trade off.

Various different approaches were followed to create multiple models. As mentioned earlier we had created multiple data sets like tree, tree_scaled, tree_smote, tree_smote_scaled, linear, linear_scaled,linear_smote, linear_smote_scaled etc.

We have also creating two generic functions which will be used to evaluate various models and also to tweak their threshold to maximize the recall.

Usage 1 – APPLY_EVAL

This is used to train the model, apply the model on test set and then output all the performance metrics like confusion matrix, Classification report, AUC curve etc.

Usage 1 – TWEAK_THRESHOLD

This is used to tweak the threshold, once the best model has been selected after hyper parameter tuning. Threshold is tweaked to maximize the recall.

Logic 2 – APPLY_EVAL

X_train, X_test,y_train & y_test are input to the function along with the model and param grid for GCV. Model is trained, tuned then validated against the test set and performance metrics are generated.

Logic 2 – TWEAK_THRESHOLD

Threshold tweaking is done by calculating performance metrics like recall for all the values of probabilities between 0 and 1, and a step size of 0.1. Threshold with best AUC score is selected.

TREE / LINEAR	ENSEMBLE MODELLING	SCALED / UNSCALED
1. We used two data sets. ○ Tree ○ Linear 2.Tree - For Tree based models like CART, Random Forest etc. 3. Linear - For distance based models like Kmeans, LDA etc.	1. Various ensemble models were also used apart from regular models. 2. Both Bagging and Boosting approaches were tried, evaluated and compared to determine the best model for our purpose.	1. Some of the models were sensitive to scaling e.g. SVM, KMeans etc. 2. On the other hand we had models like Logit and other tree based models which are scaling agnostic, we used unscaled data set there.

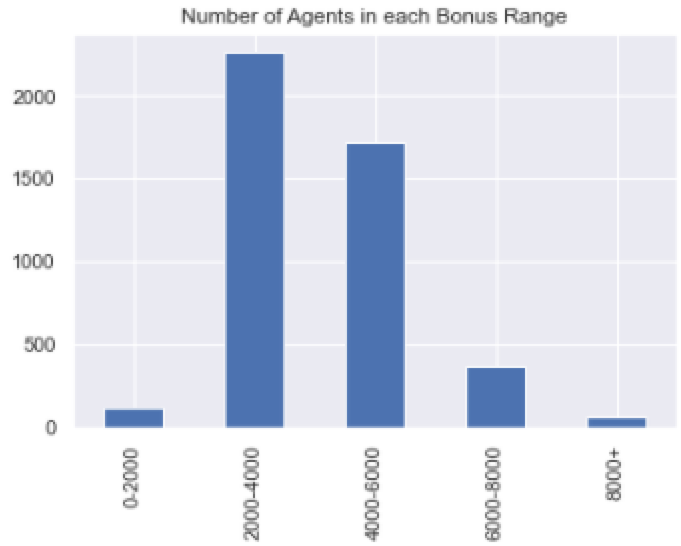
We have created multiple models as part of the Agent bonus prediction. The models include descriptive models like KMeans where we try to segment the gain insights and also predictive classification models like Random Forest, Gradient Boosting model, Logistic regression in order to predict bonus. Combined they can provide prescriptive analysis to the life insurance company and help them with the strategies.

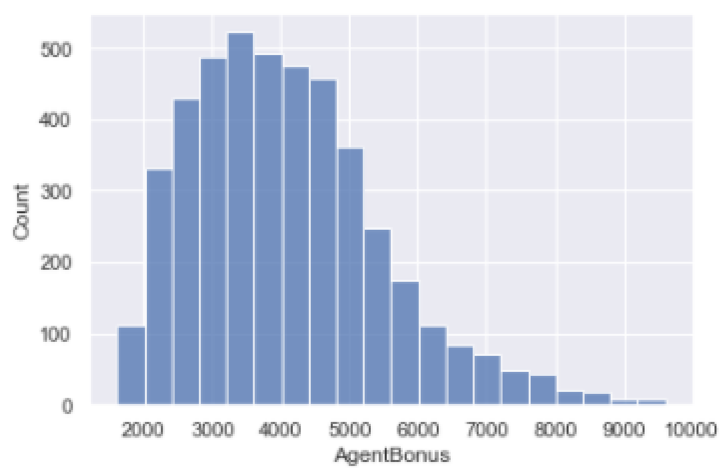
Various permutation and combinations were tried for various models.

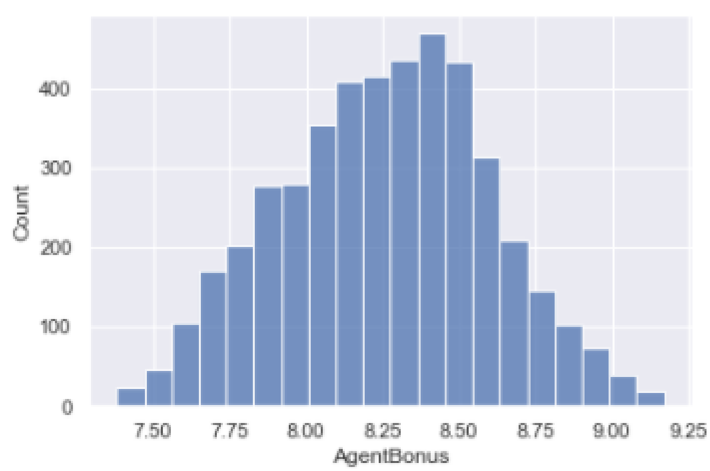
- We have included the distribution of price at different percentiles

```
In [39]: #Let's look at the distribution of price at different percentiles
print("0.5% properties have a price lower than {0: .2f}".format(np.percentile(df["AgentBonus"], 0.5)))
print("  1% properties have a price lower than {0: .2f}".format(np.percentile(df["AgentBonus"], 1)))
print("  5% properties have a price lower than {0: .2f}".format(np.percentile(df["AgentBonus"], 5)))
print(" 10% properties have a price lower than {0: .2f}".format(np.percentile(df["AgentBonus"], 10)))
print(" 90% properties have a price lower than {0: .2f}".format(np.percentile(df["AgentBonus"], 90)))
print(" 95% properties have a price lower than {0: .2f}".format(np.percentile(df["AgentBonus"], 95)))
print(" 99% properties have a price lower than {0: .2f}".format(np.percentile(df["AgentBonus"], 99)))
print("99.5% properties have a price lower than {0: .2f}".format(np.percentile(df["AgentBonus"], 99.5)))

0.5% properties have a price lower than  1755.19
  1% properties have a price lower than  1876.38
  5% properties have a price lower than  2158.00
 10% properties have a price lower than  2418.00
 90% properties have a price lower than  5917.10
 95% properties have a price lower than  6755.50
 99% properties have a price lower than  8234.44
99.5% properties have a price lower than  8757.22
```

- Let's create a range variable to understand how many records we have in different slabs





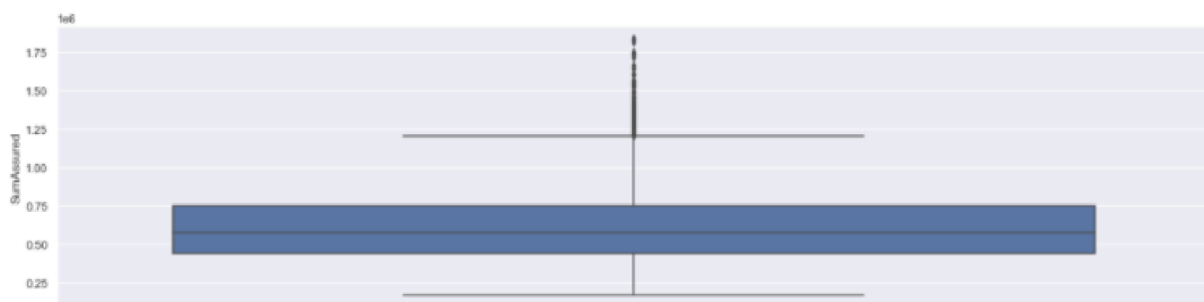


Log transformation of the AgentBonus variable looks to be slightly more symmetrically distributed. We can use a log of the AgentBonus variable as our target variable in the regression model, to check if performance is better than the AgentBonus feature used without any transformation.

Sum assured is highly correlated to Agent Bonus - we can see it in the below table.

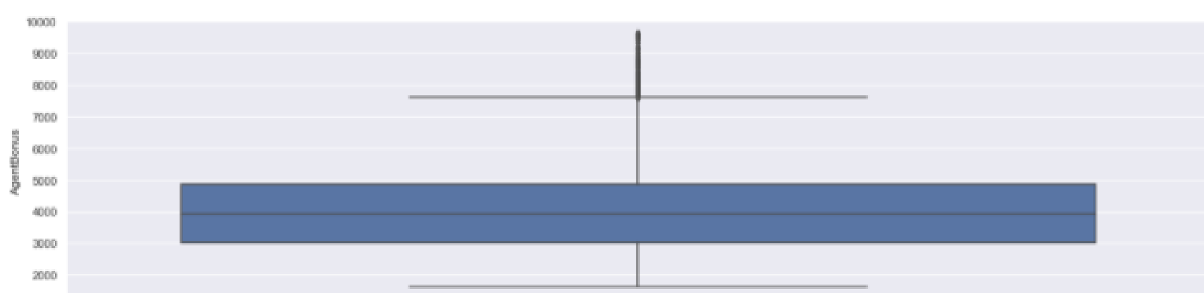
	AgentBonus	Age	CustTenure	ExistingProdType	NumberOfPolicy	MonthlyIncome	Complaint	ExistingPolicyTenure	SumAssured	LastMonthCalls	CustCareScore
AgentBonus	1	0.5523	0.5558	0.113	0.0793	0.5667	0.0143	0.3491	0.8449	0.1997	0.0232
Age	0.5523	1	0.3235	0.0735	0.0468	0.328	0.0203	0.1915	0.4662	0.1169	0.0343
CustTenure	0.5558	0.3235	1	0.0828	0.0487	0.3184	0.0043	0.1928	0.4682	0.1177	0.0115
ExistingProdType	0.113	0.0735	0.0828	1	0.1499	0.1906	-0.003	0.0593	0.1037	0.0332	0.0041
NumberOfPolicy	0.0793	0.0468	0.0487	0.1499	1	0.1335	-0.016	0.0505	0.0638	0.0751	-0.001
MonthlyIncome	0.5667	0.328	0.3184	0.1906	0.1335	1	-0.005	0.1425	0.4607	0.3374	0.0356
Complaint	0.0143	0.0203	0.0043	-0.003	-0.016	-0.005	1	0.0027	-2E-04	-0.026	-0.004
ExistingPolicyTenure	0.3491	0.1915	0.1928	0.0593	0.0505	0.1425	0.0027	1	0.3018	0.0965	-0.007
SumAssured	0.8449	0.4662	0.4682	0.1037	0.0638	0.4607	-2E-04	0.3018	1	0.158	0.0033
LastMonthCalls	0.1997	0.1169	0.1177	0.0332	0.0751	0.3374	-0.026	0.0965	0.158	1	0.0064
CustCareScore	0.0232	0.0343	0.0115	0.0041	-0.001	0.0356	-0.004	-0.007	0.0033	0.0064	1

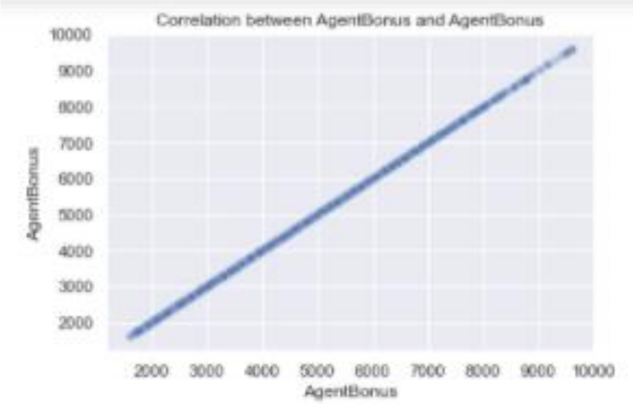
Let's check if being a SumAssured has any bonus impact



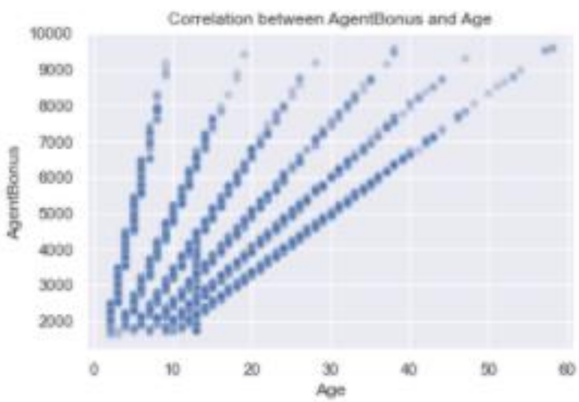

```
: plt.figure(figsize=(20,5))
  sns.boxplot(y="AgentBonus", data=df)
```

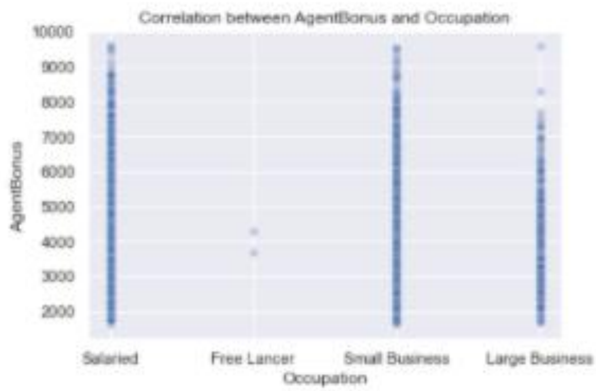
```
: <AxesSubplot:ylabel='AgentBonus'>
```

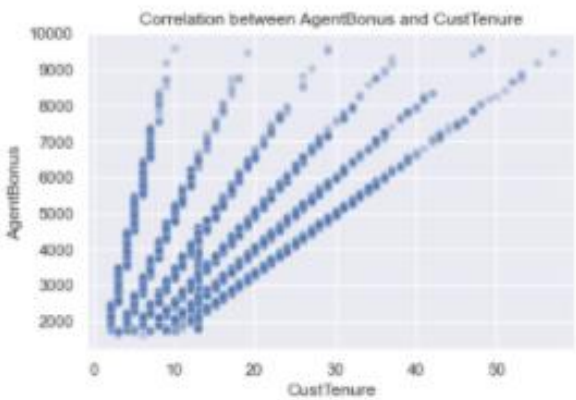


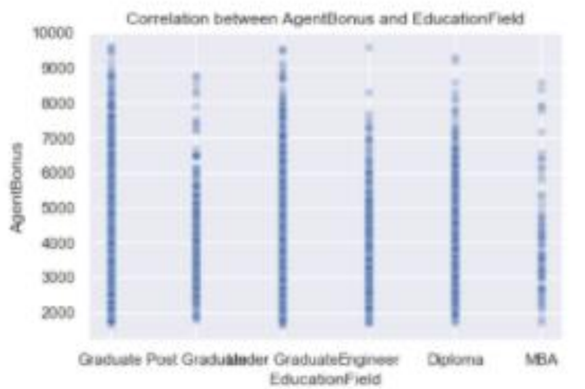

Correlation between AgentBonus and AgentBonus
AgentBonus
AgentBonus

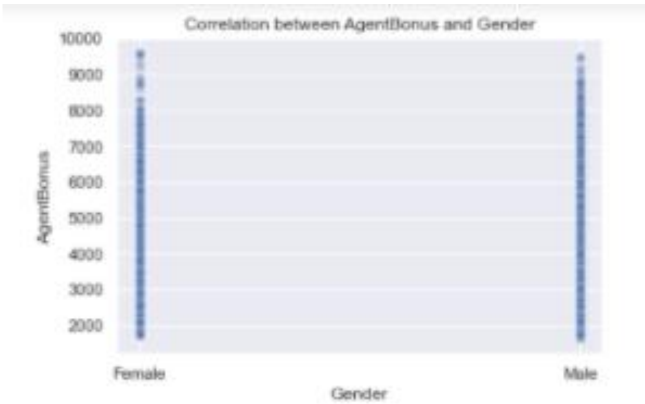

Correlation between AgentBonus and Channel
AgentBonus
Agent
Third Party Partner
Online
Channel

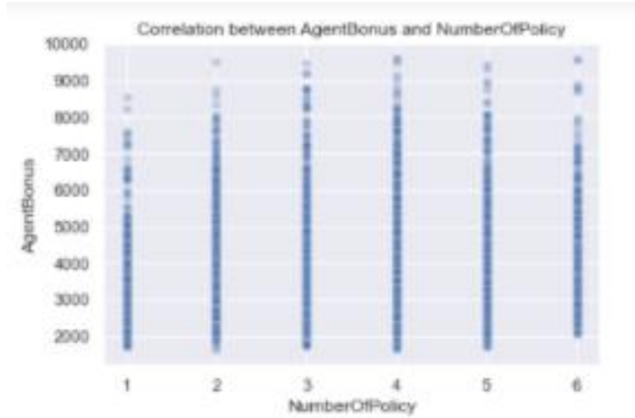

Correlation between AgentBonus and Age
AgentBonus
Age

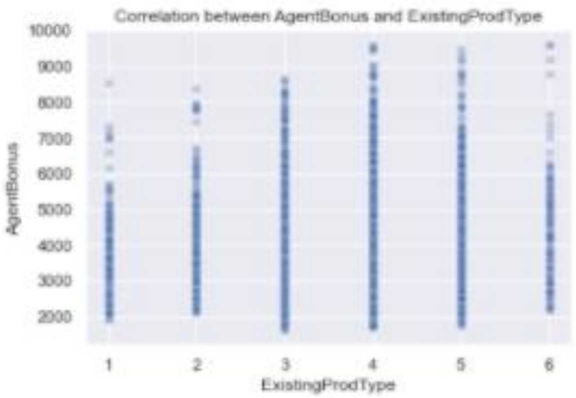

Correlation between AgentBonus and Occupation
AgentBonus
Salaried
Free Lancer
Small Business
Large Business
Occupation

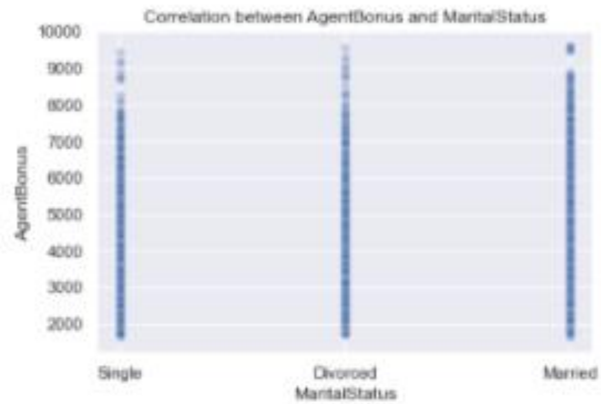

Correlation between AgentBonus and CustTenure
AgentBonus
CustTenure

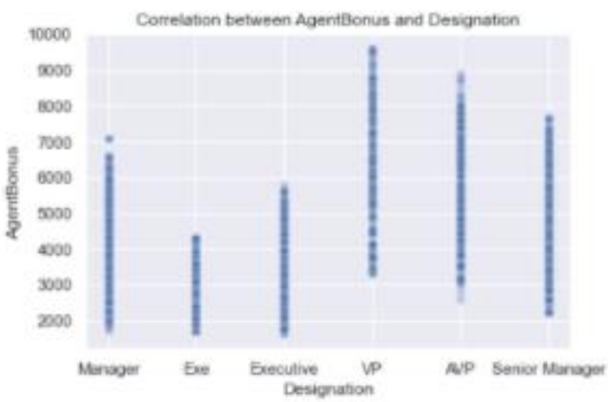

Correlation between AgentBonus and EducationField
AgentBonus
Graduate
Post Graduate
Under Graduate
Engineer
Diploma
MBA
EducationField

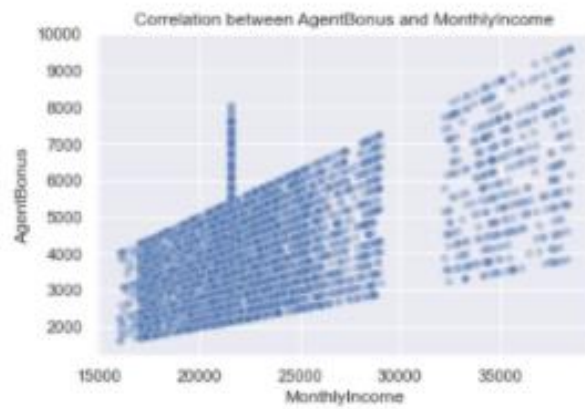

Correlation between AgentBonus and Gender
AgentBonus
Female
Male
Gender

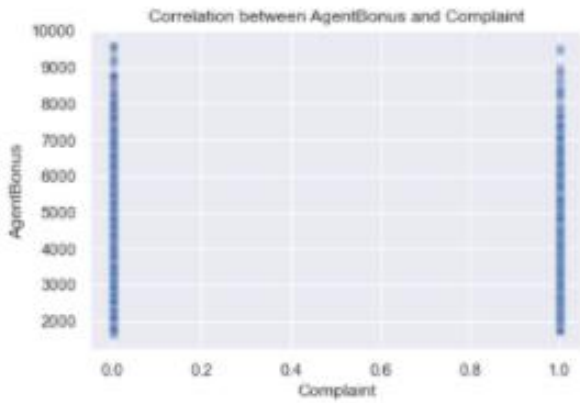

Correlation between AgentBonus and NumberOfPolicy
AgentBonus
NumberOfPolicy


Correlation between AgentBonus and ExistingProdType
AgentBonus
ExistingProdType

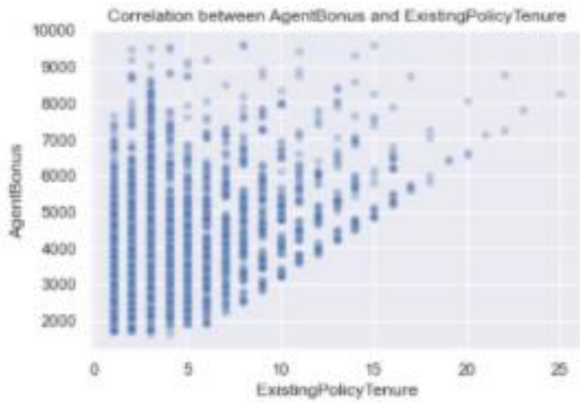

Correlation between AgentBonus and MaritalStatus
AgentBonus
Single
Divorced
Married
MaritalStatus


Correlation between AgentBonus and Designation
AgentBonus
Manager
Exe
Executive
VP
AVP
Senior Manager
Designation

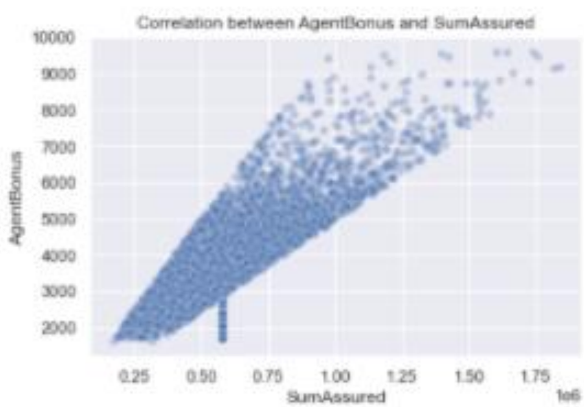

Correlation between AgentBonus and MonthlyIncome
AgentBonus
MonthlyIncome

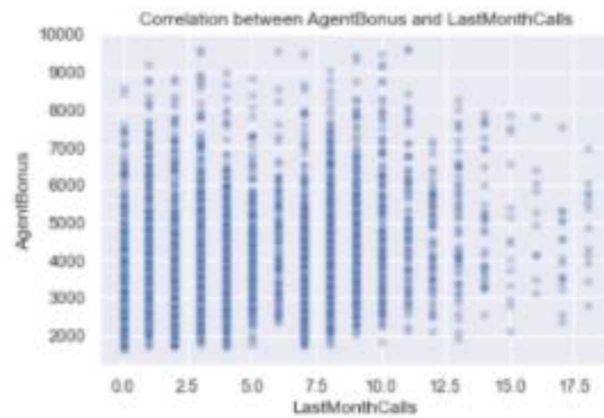

Correlation between AgentBonus and Complaint
AgentBonus
Complaint

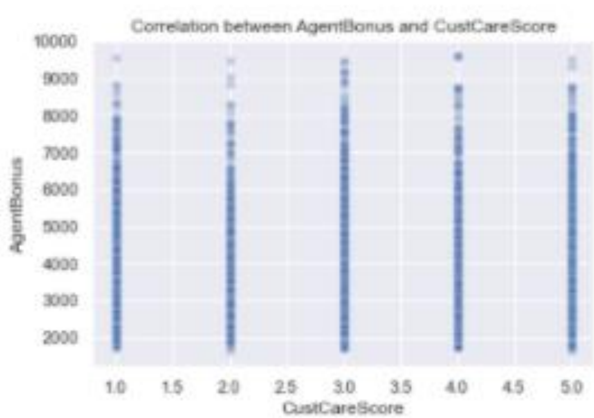

Correlation between AgentBonus and Zone
AgentBonus
North
West
East
South
Zone

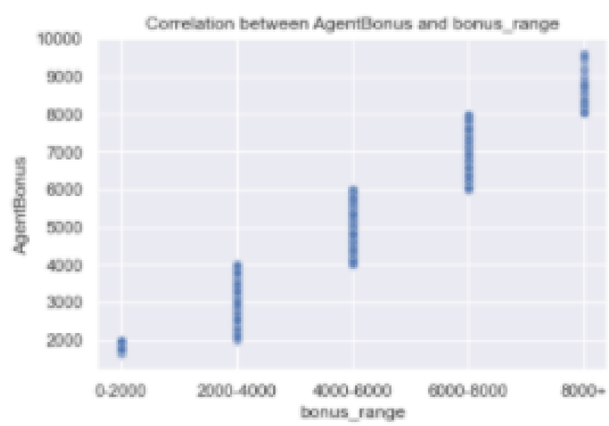

Correlation between AgentBonus and ExistingPolicyTenure
AgentBonus
ExistingPolicyTenure

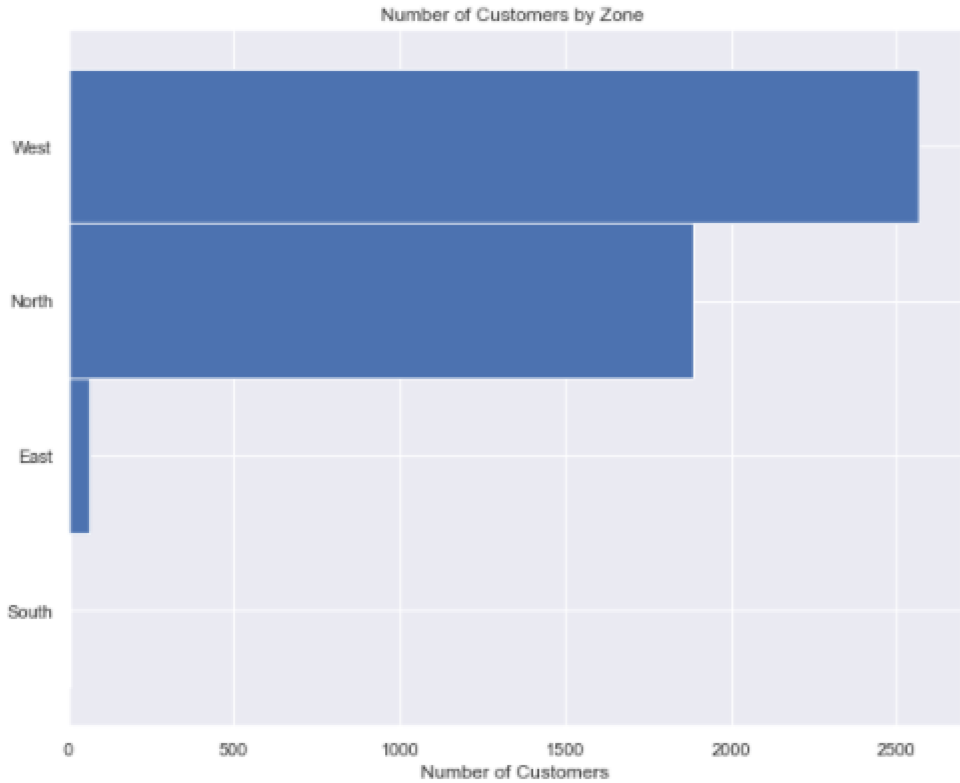

Correlation between AgentBonus and PaymentMethod
AgentBonus
Half Yearly
Yearly
Quarterly
Monthly
PaymentMethod

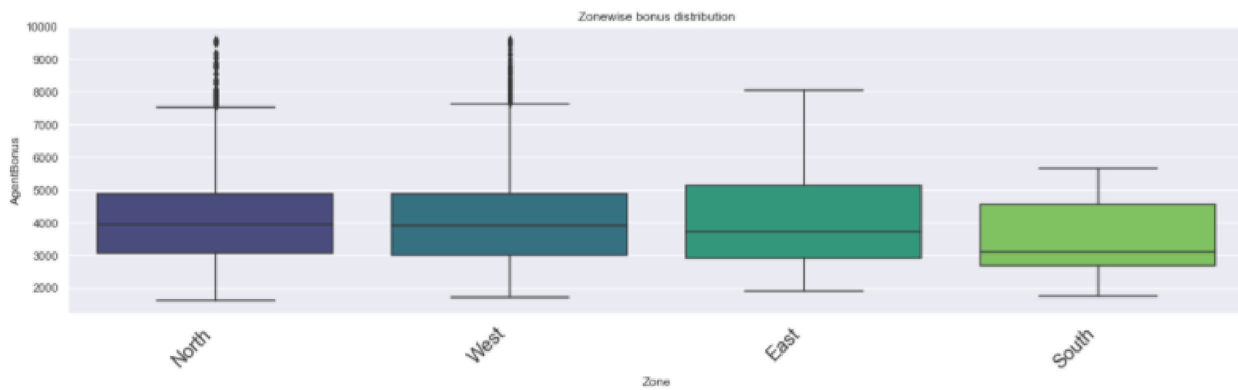

Correlation between AgentBonus and SumAssured
AgentBonus
SumAssured
1e6


Correlation between AgentBonus and LastMonthCalls
AgentBonus
LastMonthCalls


Correlation between AgentBonus and CustCareScore
AgentBonus
CustCareScore


Correlation between AgentBonus and bonus_range
AgentBonus
0-2000
2000-4000
4000-6000
6000-8000
8000+
bonus_range





Above box plots gives us the zone wise distribution

Below table shows StandardScaler

	AgentBonus	Age	CustTenure	Channel	Occupation	EducationField	Gender	ExistingProdType	Designation	NumberOfPolicy	MaritalStatus	MonthlyIncome	Complaint	ExistingPolicyTenure	SumAssured	Zone	PaymentMethod	LastMonthCalls	CustCareScore
0	4409	22	4	Agent	Salaried	Graduate	Female	3	Manager	2	Single	20993	1	2	806761	North	Half Yearly	5	2
1	2214	11	2	Third Party Partner	Salaried	Graduate	Male	4	Manager	4	Divorced	20130	0	3	294502	North	Yearly	7	3
2	4273	26	4	Agent	Free Lancer	Post Graduate	Male	4	Exe	3	Single	17090	1	2	578977	North	Yearly	0	3
3	1791	11	13	Third Party Partner	Salaried	Graduate	Female	3	Executive	3	Divorced	17909	1	2	268635	West	Half Yearly	0	5
4	2955	6	13	Agent	Small Business	Under Graduate	Male	3	Executive	4	Divorced	18468	0	4	366405	West	Half Yearly	2	5
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
4515	3953	4	8	Agent	Small Busines	Graduate	Male	4	Senior Manage	2	Single	26355	0	2	636473	West	Yearly	9	1
4516	2939	9	9	Agent	Salaried	Under Graduate	Female	2	Executive	2	Married	20991	0	3	296813	North	Yearly	1	3
4517	3792	23	23	Agent	Salaried	Engineer	Female	5	AVP	5	Single	21606	0	2	667371	North	Half Yearly	4	1
4518	4816	10	10	Online	Small Busines	Graduate	Female	4	Executive	2	Single	20068	0	6	943999	West	Half Yearly	1	5
4519	4764	14	10	Agent	Salaried	Under Graduate	Female	5	Manager	2	Married	23820	0	3	700308	North	Half Yearly	1	3

Then we apply Zscore

In [98]:
```
#for feature in cat_names:
#    if df[feature].dtype == 'object':
#        df[feature] = pd.Categorical(df[feature]).codes
df = pd.get_dummies(df, columns=cat_names,drop_first=True)
```

In [99]:
```
from scipy.stats import zscore

scaled_df= df.apply(zscore)
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4520 entries, 0 to 4519
Data columns (total 35 columns):
 #   Column                       Non-Null Count  Dtype  
---  ------                       --------------  -----  
 0   AgentBonus                   4520 non-null   int64  
 1   Age                          4520 non-null   float64
 2   CustTenure                   4520 non-null   float64
 3   ExistingProdType             4520 non-null   int64  
 4   NumberOfPolicy               4520 non-null   float64
 5   MonthlyIncome                4520 non-null   float64
 6   Complaint                    4520 non-null   int64  
 7   ExistingPolicyTenure         4520 non-null   float64
 8   SumAssured                   4520 non-null   float64
 9   LastMonthCalls               4520 non-null   int64  
 10  CustCareScore                4520 non-null   float64
 11  Channel_Online               4520 non-null   uint8  
 12  Channel_Third Party Partner  4520 non-null   uint8  
 13  Occupation_Large Business    4520 non-null   uint8  
 14  Occupation_Salaried          4520 non-null   uint8  
 15  Occupation_Small Business    4520 non-null   uint8  
 16  EducationField_Engineer      4520 non-null   uint8  
 17  EducationField_Graduate      4520 non-null   uint8  
 18  EducationField_MBA           4520 non-null   uint8  
 19  EducationField_Post Graduate 4520 non-null   uint8  
 20  EducationField_Under Graduate 4520 non-null  uint8  
 21  Gender_Male                  4520 non-null   uint8  
 22  Designation_Exe              4520 non-null   uint8  
 23  Designation_Executive        4520 non-null   uint8  
 24  Designation_Manager          4520 non-null   uint8  
 25  Designation_Senior Manager   4520 non-null   uint8  
 26  Designation_VP               4520 non-null   uint8  
 27  MaritalStatus_Married        4520 non-null   uint8  
 28  MaritalStatus_Single         4520 non-null   uint8  
 29  Zone_North                   4520 non-null   uint8  
 30  Zone_South                   4520 non-null   uint8  
 31  Zone_West                    4520 non-null   uint8  
 32  PaymentMethod_Monthly        4520 non-null   uint8  
 33  PaymentMethod_Quarterly      4520 non-null   uint8  
 34  PaymentMethod_Yearly         4520 non-null   uint8  
dtypes: float64(7), int64(4), uint8(24)
memory usage: 494.5 KB
```

New data set - scaled_df = X.fit_transform(df)

scaled_df = X.fit_transform(df)

```
In [102]: scaled_df
```

Out[102]:

	AgentBonus	Age	CustTenure	ExistingProdType	NumberOfPolicy	MonthlyIncome	Complaint	ExistingPolicyTenure	SumAssured	LastMonthCalls	C
0	0.236010	0.865868	-1.189214	-0.678318	-1.083186	-0.384155	1.575525	-0.634461	0.777226	0.103049	
1	-1.328309	-0.388311	-1.418006	0.306267	0.296941	-0.565291	-0.634709	-0.330028	-1.338756	0.655576	
2	0.139087	1.321933	-1.189214	0.306267	-0.393123	-1.203361	1.575525	-0.634461	-0.163681	-1.278269	
3	-1.629770	-0.388311	-0.159648	-0.678318	-0.393123	-1.031460	1.575525	-0.634461	-1.445604	-1.278269	
4	-0.800217	-0.958393	-0.159648	-0.678318	0.296941	-0.914131	-0.634709	-0.025594	-1.041747	-0.725742	
...	...	...	...	...	...	...	...	...	...	...	
4515	-0.088969	-1.186425	-0.731629	0.306267	-1.083186	0.741284	-0.634709	-0.634461	0.073819	1.208103	
4516	-0.811620	-0.616344	-0.617233	-1.662902	-1.083186	-0.384574	-0.634709	-0.330028	-1.329210	-1.002006	
4517	-0.203709	0.979884	0.984313	1.290851	0.987005	-0.255491	-0.634709	-0.634461	0.201449	-0.173215	
4518	0.526069	-0.502328	-0.502837	0.306267	-1.083186	-0.578304	-0.634709	0.583273	1.344113	-1.002006	
4519	0.489009	-0.046262	-0.502837	1.290851	-1.083186	0.209209	-0.634709	-0.330028	0.337502	-1.002006	

4520 rows × 35 columns

In [103]:
```
df
```
Out[103]:

	AgentBonus	Age	CustTenure	ExistingProdType	NumberOfPolicy	MonthlyIncome	Complaint	ExistingPolicyTenure	SumAssured	LastMonthCalls	CustC
0	4409	22.0	4.0	3	2.0	20993.0	1	2.0	806761.0	5	
1	2214	11.0	2.0	4	4.0	20130.0	0	3.0	294502.0	7	
2	4273	26.0	4.0	4	3.0	17090.0	1	2.0	578976.5	0	
3	1791	11.0	13.0	3	3.0	17909.0	1	2.0	268635.0	0	
4	2955	6.0	13.0	3	4.0	18468.0	0	4.0	366405.0	2	
...	...	...	...	...	...	...	...	...	...	...	
4515	3953	4.0	8.0	4	2.0	26355.0	0	2.0	636473.0	9	
4516	2939	9.0	9.0	2	2.0	20991.0	0	3.0	296813.0	1	
4517	3792	23.0	23.0	5	5.0	21606.0	0	2.0	667371.0	4	
4518	4816	10.0	10.0	4	2.0	20068.0	0	6.0	943999.0	1	
4519	4764	14.0	10.0	5	2.0	23820.0	0	3.0	700308.0	1	

4520 rows × 35 columns

We create the Covariance Matrix

```
Covariance Matrix
%s [[ 1.00022129e+00  5.52466509e-01  5.55914247e-01 ... -8.68720331e-0
  -8.74510181e-03 -8.18575787e-03]
 [ 5.52466509e-01  1.00022129e+00  3.23557414e-01 ...  8.79326307e-04
   7.68499034e-03  7.38095236e-03]
 [ 5.55914247e-01  3.23557414e-01  1.00022129e+00 ... -1.14980907e-02
  -1.97010911e-02 -3.93008587e-03]
 ...
 [-8.68720331e-03  8.79326307e-04 -1.14980907e-02 ...  1.00022129e+00
  -3.81292149e-02 -1.98753464e-01]
 [-8.74510181e-03  7.68499034e-03 -1.97010911e-02 ... -3.81292149e-02
   1.00022129e+00 -8.91646028e-02]
 [-8.18575787e-03  7.38095236e-03 -3.93008587e-03 ... -1.98753464e-01
  -8.91646028e-02  1.00022129e+00]]
```

Step 2- Get eigen values and eigen vector

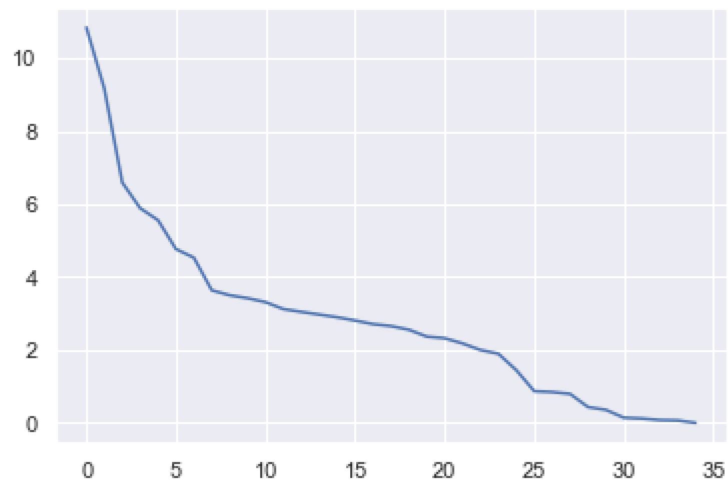
```
 Eigen Values
 %s [3.79451749e+00 3.20855610e+00 2.31059807e+00 2.06341378e+00
 1.94793092e+00 1.67042542e+00 1.58802281e+00 7.49061134e-04
 2.46621967e-02 2.80642484e-02 5.03018207e-02 4.28193698e-02
 1.25809919e-01 1.52751083e-01 2.79123158e-01 3.05317918e-01
 2.97177481e-01 5.07059653e-01 6.64463202e-01 1.27461957e+00
 7.00645098e-01 1.22669205e+00 1.19838224e+00 1.16053227e+00
 7.63975356e-01 8.13956286e-01 8.29716857e-01 1.09375028e+00
 8.95725076e-01 1.06691596e+00 9.30864626e-01 9.49509337e-01
 1.04123756e+00 1.01539374e+00 9.84065068e-01]
Eigen Vectors
 %s [[ 0.45237219  0.01517699 -0.0181631  ... -0.01205252 -0.02722782
   0.00392377]
 [ 0.31195074  0.01587582 -0.01516285 ... -0.0342978  -0.02133003
  -0.0397793 ]
 [ 0.31138038  0.01361653 -0.00718656 ...  0.01256985 -0.07818545
  -0.00172299]
 ...
 [-0.0387375   0.0036899   0.16576068 ...  0.17932336 -0.03923713
   0.10230862]
 [-0.01344512  0.00076899  0.06259944 ... -0.61504789  0.24511162
  -0.23502232]
 [ 0.02441971  0.00740794 -0.15300843 ...  0.0672648   0.02552505
   0.01548285]]
```

We also performed Cumulative Variance

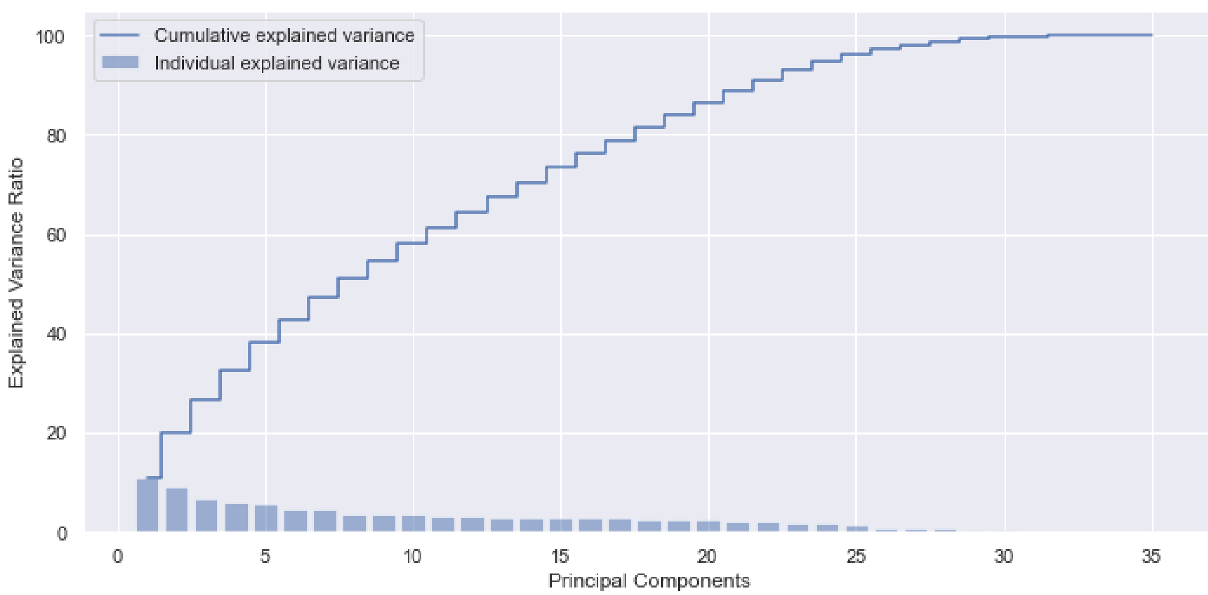
```
Cumulative Variance Explained [ 10.83907998  20.00435496  26.60460318  32.49876681  38.06305242
  42.83464059  47.3708448   51.01180929  54.51586831  57.93906005
  61.25413296  64.3784424   67.4260993   70.40040559  73.30088872
  76.11188116  78.82416479  81.48318959  84.0418379   86.41193302
  88.73700789  90.91931171  92.92071196  94.81875824  96.26717958
  97.1393235   97.98821417  98.78553246  99.22186758  99.58124496
  99.72493265  99.84724664  99.92741247  99.9978603  100.        ]
```

In [107]:
```
plt.plot(var_exp)
```

Out[107]: [<matplotlib.lines.Line2D at 0x2229cd2a8b0>]



The below figure shows Individual explained variance and Cumulative explained variance plotted against Explained Variance Ratio Principal Components



Then using scikit learn PCA. It does all the above steps and maps data to PCA dimensions in one shot

NOTE - we are generating only 4 PCA dimensions (dimensionality reduction from 18 to 4)

```
array([[-0.15534127, -1.81152561, -1.31722085, ...,  0.37720728,
        -0.345225  ,  0.26753463],
       [-1.88925042, -1.83007197, -0.41906599, ..., -0.69035217,
         0.08630408,  0.0882461 ],
       [-0.69077934, -0.51846599,  0.35539171, ...,  2.2400937 ,
         0.08296675, -0.96193646],
       [ 0.89252094,  0.19278019, -0.11822886, ...,  0.96084063,
        -1.6790074 ,  0.86128105],
       [-0.54629398, -1.90203475, -1.85047993, ..., -1.45433386,
         0.39490144, -1.52215013]])
```

```
In [111]: pca.explained_variance_ratio_
Out[111]: array([0.10839011, 0.09165256, 0.06595155, 0.05892737, 0.05557526])
```

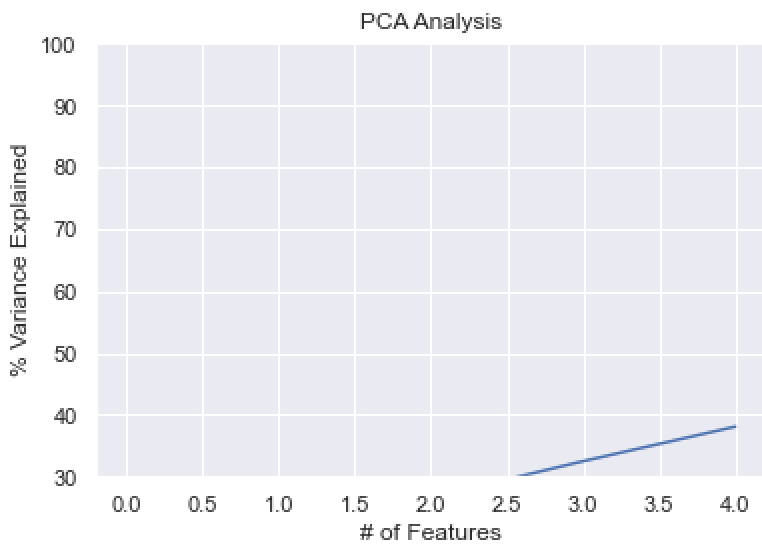
Visually we can observe that there is steep drop in variance explained with increase in number of PC's.

We will proceed with 5 components here. But depending on requirement 90% variation or 5 components will also do well.

Cumulative sum of variance explained with [n] features

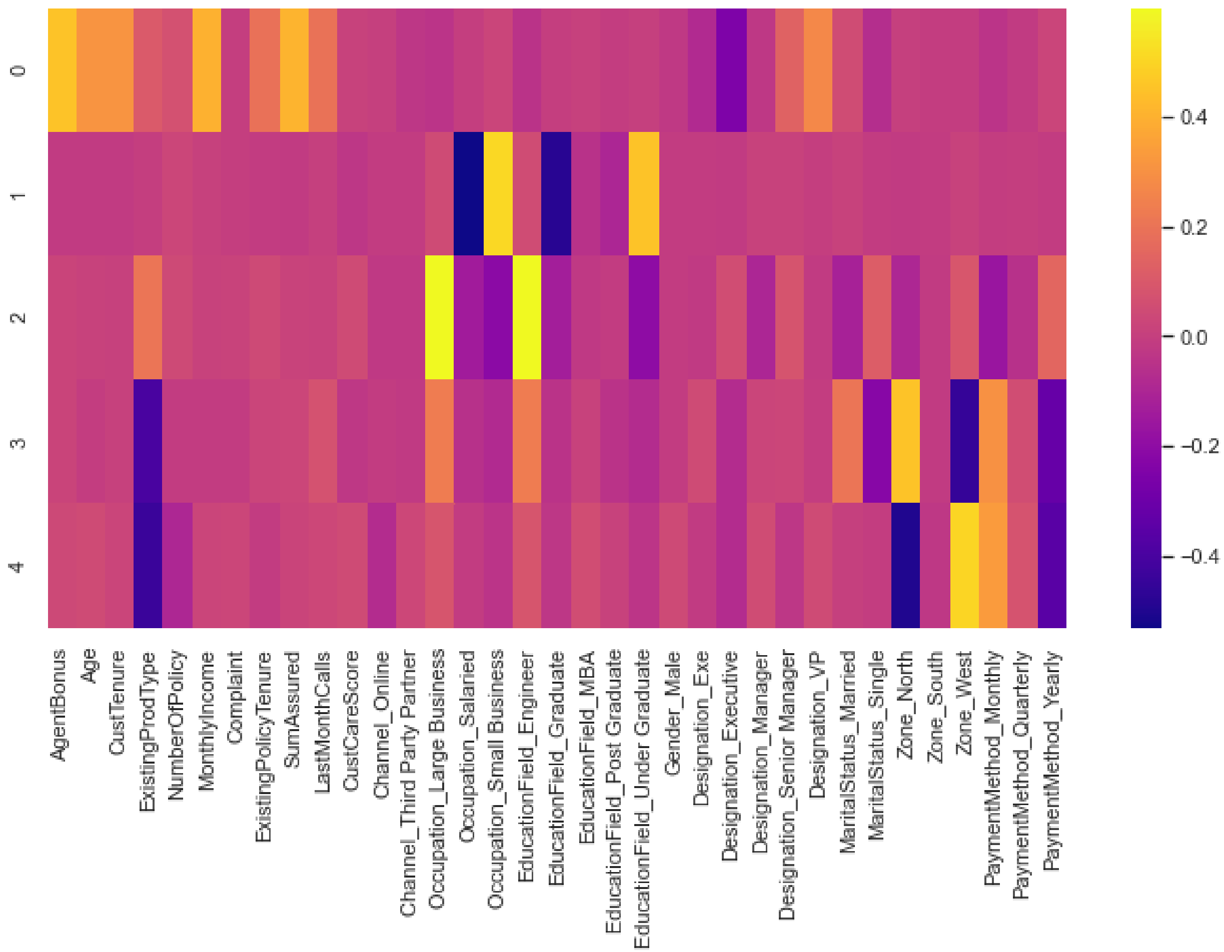
```
Out[113]: array([10.8, 20. , 26.6, 32.5, 38.1])
```

Below graph shows the PCA Analysis.



Below table gives the snapshot of scaled data frame

	AgentBonus	Age	CustTenure	ExistingProdType	NumberOfPolicy	MonthlyIncome	Complaint	ExistingPolicyTenure	SumAssured	LastMonthCalls
0	0.452304	0.312155	0.311311	0.107020	0.069352	0.401866	-0.001750	0.194515	0.410951	0.196427
1	-0.015104	-0.015866	-0.013475	-0.000535	0.032386	0.008376	-0.002748	-0.010990	-0.012722	0.005030
2	0.019500	0.013631	0.007669	0.207772	0.043811	0.012975	0.016080	0.041373	0.015310	0.011347
3	0.022079	-0.005888	0.008401	-0.397152	-0.007238	-0.007332	-0.007848	0.028052	0.029399	0.074443
4	0.037635	0.045849	0.027682	-0.438780	-0.091263	0.023763	0.030426	-0.010050	0.020139	0.033459



The above heat map and the color bar basically represent the correlation between the various features and the principal component itself. Component 2 looks more related to aspect - We can label it as aspect property. Depending on relations ship, we could go ahead and label relationship with features.

PROBLEM 1.B

Test your predictive model against the test set using various appropriate performance metrics

Resolution:

Here we start with KMeans clustering

In [119]:
```
k_means = KMeans(n_clusters = 2)
```

In [120]:
```
k_means.fit(scaled_df)
```

Out[120]:
```
KMeans(n_clusters=2)
```

In [121]:
```
k_means.labels_
```

Out[121]:
```
array([1, 1, 1, ..., 1, 0, 1])
```

In [122]:
```
k_means.inertia_
```

Out[122]:
```
144404.42847886533
```

In [123]:
```
k_means = KMeans(n_clusters = 3)
k_means.fit(scaled_df)
k_means.inertia_
```

Out[123]:
```
134196.70230079163
```

In [124]:
```
k_means = KMeans(n_clusters = 4)
k_means.fit(scaled_df)
k_means.inertia_
```

Out[124]:
```
126689.45524508599
```

In [125]:
```
k_means = KMeans(n_clusters = 1)
k_means.fit(scaled_df)
k_means.inertia_
```

Out[125]:
```
158200.00000000017
```

In [126]:
```
k_means = KMeans(n_clusters = 5)
k_means.fit(scaled_df)
k_means.inertia_
```

Out[126]:
```
121886.96064843437
```

In [127]:
```
k_means = KMeans(n_clusters = 6)
k_means.fit(scaled_df)
k_means.inertia_
```

Out[127]:
```
117897.29294434794
```

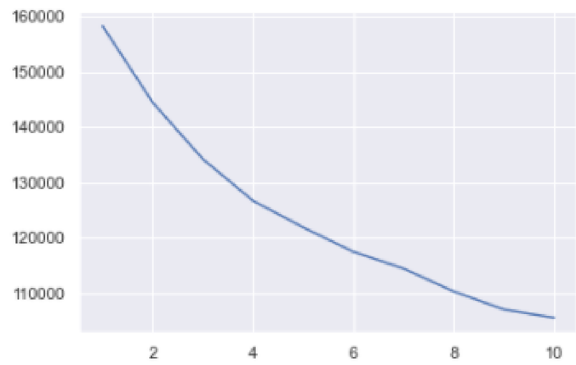
In [128]:
```
wss =[]
```

In [129]:
```
for i in range(1,11):
    KM = KMeans(n_clusters=i)
    KM.fit(scaled_df)
    wss.append(KM.inertia_)
```

In [130]:
```
wss
```

Out[130]:
```
[158200.00000000017,
 144404.42847886533,
 134196.70230079163,
 126689.45524508599,
 121886.96064843437,
 117490.11982618901,
 114471.98249825888,
 110332.58417396643,
 107078.37139779047,
 105590.47261570446]
```

```
In [131]: plt.plot(range(1,11), wss)
Out[131]: [<matplotlib.lines.Line2D at 0x2229c6582b0>]
```



The above graph shows the WSS

```
df["Clus_kmeans"] = labels
df.head(5)
```

	AgentBonus	Age	CustTenure	ExistingProdType	NumberOfPolicy	MonthlyIncome	Complaint	ExistingPolicyTenure	SumAssured	LastMonthCalls	CustCareS
0	4409	22.0	4.0	3	2.0	20993.0	1	2.0	806761.0	5	
1	2214	11.0	2.0	4	4.0	20130.0	0	3.0	294502.0	7	
2	4273	26.0	4.0	4	3.0	17090.0	1	2.0	578976.5	0	
3	1791	11.0	13.0	3	3.0	17909.0	1	2.0	268635.0	0	
4	2955	6.0	13.0	3	4.0	18468.0	0	4.0	366405.0	2	

After the clustering we prepare

```
from scipy.cluster.hierarchy import dendrogram, linkage
```

```
link_method = linkage(scaled_df.iloc[:,1:6], method = 'average')
```

```
link_method
```

```
array([[4.48000000e+02, 6.58000000e+02, 0.00000000e+00, 2.00000000e+00],
       [4.84000000e+02, 6.13000000e+02, 0.00000000e+00, 2.00000000e+00],
       [1.02000000e+02, 1.57200000e+03, 0.00000000e+00, 2.00000000e+00],
       ...,
       [9.03200000e+03, 9.03500000e+03, 4.58893877e+00, 4.51100000e+03],
       [4.27600000e+03, 9.03600000e+03, 4.78603289e+00, 4.51200000e+03],
       [9.02500000e+03, 9.03700000e+03, 6.99203394e+00, 4.52000000e+03]])
```

```
labelList = np.array(scaled_df.AgentBonus)
labelList
```

```
array([ 0.23601029, -1.32830873,  0.13908665, ..., -0.20370945,
        0.52606853,  0.4890095 ])
```

Now we create Regression Model

```
In [141]: from sklearn.feature_selection import RFE
          from sklearn.model_selection import train_test_split
          from sklearn.linear_model import LinearRegression

          Y = df[["AgentBonus"]]
          X = df.drop("AgentBonus", axis=1)
          X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=0.30 , random_state=8)
          from scipy.stats import zscore

          X_train_scaled  = X_train.apply(zscore)
          X_test_scaled = X_test.apply(zscore)
          y_train_scaled = y_train.apply(zscore)
          y_test_scaled = y_test.apply(zscore)
```

```
selector.n_features_
```

```
15
```

```
selector.ranking_
```

```
array([11,  9,  1, 13, 20, 19,  8, 21, 17, 16, 12, 15,  1,  1,  1,  1,  1,
        7,  2,  4, 18,  1,  1,  1,  1,  3, 10, 14,  5,  1,  6,  1,  1,  1,
        1])
```

Below table gives us the feature and the rank

	Feature	Rank
2	ExistingProdType	1
12	Occupation_Large Business	1
13	Occupation_Salaried	1
14	Occupation_Small Business	1
15	EducationField_Engineer	1
16	EducationField_Graduate	1
21	Designation_Exe	1
22	Designation_Executive	1
23	Designation_Manager	1
24	Designation_Senior Manager	1
29	Zone_South	1
31	PaymentMethod_Monthly	1
32	PaymentMethod_Quarterly	1
33	PaymentMethod_Yearly	1
34	Clus_kmeans	1

Most important features are Designation, Occupation, South zone, Payment Method, Existing Prod Type, Education Field,

Zone west and North, Existing Policy Tenure, Cust Tenure, Age

We have also created Agglomerative Clustering

```
cluster = AgglomerativeClustering(n_clusters=5, affinity='euclidean', linkage='average')
Cluster_agglo=cluster.fit_predict(scaled_df.iloc[:,1:6])
print(Cluster_agglo)
```

```
[4 4 4 ... 4 4 4]
```

We then drop df.drop(columns=['Agglo_CLusters'],inplace=True)

```
Index(['AgentBonus', 'Age', 'CustTenure', 'ExistingProdType', 'NumberOfPolicy',
       'MonthlyIncome', 'Complaint', 'ExistingPolicyTenure', 'SumAssured',
       'LastMonthCalls', 'CustCareScore', 'Channel_Online',
       'Channel_Third Party Partner', 'Occupation_Large Business',
       'Occupation_Salaried', 'Occupation_Small Business',
       'EducationField_Engineer', 'EducationField_Graduate',
       'EducationField_MBA', 'EducationField_Post Graduate',
       'EducationField_Under Graduate', 'Gender_Male', 'Designation_Exe',
       'Designation_Executive', 'Designation_Manager',
       'Designation_Senior Manager', 'Designation_VP', 'MaritalStatus_Married',
       'MaritalStatus_Single', 'Zone_North', 'Zone_South', 'Zone_West',
       'PaymentMethod_Monthly', 'PaymentMethod_Quarterly',
       'PaymentMethod_Yearly', 'Clus_kmeans'],
      dtype='object')
```

Below table shows the grouping by Agglo_Clusters

Agglo_Clusters	0	1	2	3	4
AgentBonus	6132.635922	6741.50	7468.0	9195.375	3857.676679
Age	22.885922	9.75	7.0	55.250	13.479121
CustTenure	22.283981	39.00	45.0	41.250	13.517949
ExistingProdType	3.783981	6.00	2.0	4.625	3.675702
NumberOfPolicy	3.774272	4.00	6.0	4.250	3.546764
MonthlyIncome	34274.157767	29199.00	21606.0	37733.750	21636.112088
Complaint	0.283981	0.00	1.0	0.250	0.287668
ExistingPolicyTenure	4.893204	7.75	4.0	4.625	3.998046
SumAssured	907792.424757	1089903.25	955936.0	1457508.375	587324.779731
LastMonthCalls	6.817961	4.25	8.0	6.875	4.401709
CustCareScore	3.269417	3.00	1.0	3.500	3.046154
Channel_Online	0.089806	0.00	0.0	0.250	0.104762
Channel_Third Party Partner	0.145631	0.00	0.0	0.000	0.194872
Occupation_Large Business	0.043689	0.25	0.0	0.000	0.094994
Occupation_Salaried	0.500000	0.50	1.0	0.625	0.483028
Occupation_Small Business	0.456311	0.25	0.0	0.375	0.421490
EducationField_Engineer	0.041262	0.25	0.0	0.000	0.095238
EducationField_Graduate	0.419903	0.50	1.0	0.625	0.412454
EducationField_MBA	0.036408	0.00	0.0	0.000	0.014408
EducationField_Post Graduate	0.041262	0.00	0.0	0.000	0.057387
EducationField_Under Graduate	0.354369	0.00	0.0	0.375	0.310379
Gender_Male	0.626214	0.50	0.0	0.625	0.591697
Designation_Exe	0.000000	0.00	0.0	0.000	0.031013

```
agglo_data
```

Agglo_Clusters	AgentBonus	Age	CustTenure	ExistingProdType	NumberOfPolicy	MonthlyIncome	Complaint	ExistingPolicyTenure	SumAssured	L
0	6132.635922	22.885922	22.283981	3.783981	3.774272	34274.157767	0.283981	4.893204	9.077924e+05	
1	6741.500000	9.750000	39.000000	6.000000	4.000000	29199.000000	0.000000	7.750000	1.089903e+06	
2	7468.000000	7.000000	45.000000	2.000000	6.000000	21606.000000	1.000000	4.000000	9.559360e+05	
3	9195.375000	55.250000	41.250000	4.625000	4.250000	37733.750000	0.250000	4.625000	1.457508e+06	
4	3857.676679	13.479121	13.517949	3.675702	3.546764	21636.112088	0.287668	3.998046	5.873248e+05	

Agglo_Clusters	0	1	2	3	4
AgentBonus	6132.635922	6741.50	7468.0	9195.375	3857.676679
SumAssured	907792.424757	1089903.25	955936.0	1457508.375	587324.779731
Age	22.885922	9.75	7.0	55.250	13.479121
CustTenure	22.283981	39.00	45.0	41.250	13.517949
ExistingProdType	3.783981	6.00	2.0	4.625	3.675702
MonthlyIncome	34274.157767	29199.00	21606.0	37733.750	21636.112088
Occupation_Large Business	0.043689	0.25	0.0	0.000	0.094994
Occupation_Salaried	0.500000	0.50	1.0	0.625	0.483028
Occupation_Small Business	0.456311	0.25	0.0	0.375	0.421490
EducationField_Engineer	0.041262	0.25	0.0	0.000	0.095238
EducationField_Graduate	0.419903	0.50	1.0	0.625	0.412454
EducationField_MBA	0.036408	0.00	0.0	0.000	0.014408
EducationField_Post Graduate	0.041262	0.00	0.0	0.000	0.057387
EducationField_Under Graduate	0.354369	0.00	0.0	0.375	0.310379
Designation_Manager	0.007282	0.50	0.0	0.000	0.394383
Designation_Senior Manager	0.038835	0.25	1.0	0.000	0.160684
Designation_VP	0.526699	0.25	0.0	1.000	0.000000
MaritalStatus_Married	0.550971	0.25	1.0	0.375	0.497192
Zone_South	0.000000	0.00	0.0	0.000	0.001465
PaymentMethod_Monthly	0.072816	0.00	1.0	0.000	0.078877
PaymentMethod_Quarterly	0.012136	0.00	0.0	0.000	0.017338
PaymentMethod_Yearly	0.274272	1.00	0.0	0.375	0.320879
Freq	412.000000	4.00	1.0	8.000	4095.000000
CustTenure	22.283981	39.00	45.0	41.250	13.517949
Zone_North	0.368932	0.00	1.0	0.375	0.421978
Zone_West	0.611650	1.00	0.0	0.625	0.562882
Designation_Exe	0.000000	0.00	0.0	0.000	0.031013
Designation_Executive	0.000000	0.00	0.0	0.000	0.374847

Now we get the silhouette score

```
In [159]: silhouette_score(scaled_df,labels)
Out[159]: 0.11812323239864359
```

Below table shows the data with Sil_width

	AgentBonus	Age	CustTenure	ExistingProdType	NumberOfPolicy	MonthlyIncome	Complaint	ExistingPolicyTenure	SumAssured	LastMonthCalls
0	4409	22.0	4.0	3	2.0	20993.0	1	2.0	806761.0	5
1	2214	11.0	2.0	4	4.0	20130.0	0	3.0	294502.0	7
2	4273	26.0	4.0	4	3.0	17090.0	1	2.0	578976.5	0
3	1791	11.0	13.0	3	3.0	17909.0	1	2.0	268635.0	0
4	2955	6.0	13.0	3	4.0	18468.0	0	4.0	366405.0	2

```
silhouette_samples(scaled_df,labels).min()
```

```
-0.1084158537063405
```

Next we calculate variance inflation factor

	variables	VIF
14	Occupation_Salaried	163.510600
5	MonthlyIncome	143.287388
15	Occupation_Small_Business	121.653120
3	ExistingProdType	64.107943
0	AgentBonus	49.676191
35	Agglo_Clusters	49.216890
13	Occupation_Large_Business	45.003817
31	Zone_West	40.643026
17	EducationField_Graduate	31.052427
29	Zone_North	30.124808
8	SumAssured	27.123217
16	EducationField_Engineer	20.769377
23	Designation_Executive	12.938618
24	Designation_Manager	11.160432
4	NumberOfPolicy	7.832282
10	CustCareScore	6.092959
2	CustTenure	5.448530
1	Age	5.384959
19	EducationField_Post_Graduate	5.132235
25	Designation_Senior_Manager	4.654740
20	EducationField_Under_Graduate	3.997850
27	MaritalStatus_Married	3.895606
34	PaymentMethod_Yearly	3.246193
9	LastMonthCalls	3.186624
7	ExistingPolicyTenure	2.960278
28	MaritalStatus_Single	2.871670
21	Gender_Male	2.535430
32	PaymentMethod_Monthly	2.491524
26	Designation_VP	2.364060
18	EducationField_MBA	2.250673
22	Designation_Exe	2.129217
6	Complaint	1.415376
12	Channel_Third_Party_Partner	1.285194
11	Channel_Online	1.168744
33	PaymentMethod_Quarterly	1.124055
30	Zone_South	1.097750

Here, we see that the value of VIF is high for many variables. Here, we may drop variables with VIF more than 5 (very high correlation) & build our model

Now we split the data

```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error
import statsmodels.api as sm
```

```
X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=0.30 , random_state=8)
```

```
from scipy.stats import zscore

X_train_scaled  = X_train.apply(zscore)
X_test_scaled = X_test.apply(zscore)
y_train_scaled = y_train.apply(zscore)
y_test_scaled = y_test.apply(zscore)
```

Using Statsmodels OLS

```
model = sm.OLS(y_trainc,X_trainc).fit()
model.summary()
```

OLS Regression Results

Dep. Variable:	AgentBonus	**R-squared:**	0.808
Model:	OLS	**Adj. R-squared:**	0.806
Method:	Least Squares	**F-statistic:**	376.5
Date:	Sun, 16 Jan 2022	**Prob (F-statistic):**	0.00
Time:	13:11:37	**Log-Likelihood:**	-24798.
No. Observations:	3164	**AIC:**	4.967e+04
Df Residuals:	3128	**BIC:**	4.989e+04
Df Model:	35		
Covariance Type:	nonrobust		

Covariance Type:	nonrobust

	coef	std err	t	P>|t|	[0.025	0.975]
const	1302.7182	495.358	2.630	0.009	331.458	2273.978
Age	22.3054	1.470	15.170	0.000	19.423	25.188
CustTenure	23.5837	1.459	16.165	0.000	20.723	26.444
ExistingProdType	37.0125	22.561	1.641	0.101	-7.223	81.248
NumberOfPolicy	3.6064	8.017	0.450	0.653	-12.112	19.325
MonthlyIncome	0.0322	0.006	5.443	0.000	0.021	0.044
Complaint	29.6882	24.275	1.223	0.221	-17.909	77.285
ExistingPolicyTenure	32.7568	3.527	9.288	0.000	25.842	39.672
SumAssured	0.0034	6.05e-05	56.775	0.000	0.003	0.004
LastMonthCalls	-1.7463	3.352	-0.521	0.602	-8.318	4.825
CustCareScore	3.4902	8.087	0.432	0.666	-12.365	19.346
Channel Online	15.2347	36.443	0.418	0.676	-56.221	86.690
Channel Third Party Partner	19.2115	28.375	0.677	0.498	-36.424	74.847
Occupation Large Business	-636.7982	467.077	-1.363	0.173	-1552.606	279.009
Occupation Salaried	-674.8400	441.006	-1.530	0.126	-1539.531	189.851
Occupation Small Business	-692.2619	450.606	-1.536	0.125	-1575.775	191.251
EducationField Engineer	-22.7474	172.628	-0.132	0.895	-361.222	315.727
EducationField Graduate	-8.0384	97.894	-0.082	0.935	-199.981	183.904
EducationField MBA	-148.5732	131.765	-1.127	0.260	-406.968	109.822
EducationField Post Graduate	-40.9633	107.961	-0.379	0.704	-252.645	170.719
EducationField Under Graduate	0.7167	39.936	0.018	0.986	-77.586	79.019
Gender Male	9.9057	22.637	0.438	0.662	-34.479	54.290
Designation Exe	-943.0769	96.711	-9.752	0.000	-1132.700	-753.454
Designation Executive	-482.5651	68.032	-7.093	0.000	-615.958	-349.174
Designation Manager	-478.3649	61.472	-7.782	0.000	-598.894	-357.835
Designation Senior Manager	-288.3461	62.548	-4.610	0.000	-410.986	-165.706
Designation VP	112.3353	74.772	1.502	0.133	-34.271	258.942
MaritalStatus Married	0.3029	30.332	0.010	0.992	-59.169	59.775
MaritalStatus Single	5.7490	32.664	0.176	0.860	-58.297	69.795
Zone North	-83.1301	96.303	-0.863	0.388	-271.953	105.693
Zone South	-174.9996	369.620	-0.473	0.636	-900.114	550.115
Zone West	-79.1350	95.765	-0.826	0.409	-266.942	108.673
PaymentMethod Monthly	149.5088	61.407	2.435	0.015	29.106	269.910
PaymentMethod Quarterly	132.0691	90.469	1.460	0.145	-45.355	309.493
PaymentMethod Yearly	-58.4653	34.522	-1.694	0.090	-126.154	9.223
Agglo Clusters	35.2088	21.527	1.636	0.102	-7.000	77.418

Omnibus:	190.852	Durbin-Watson:	2.027
Prob(Omnibus):	0.000	Jarque-Bera (JB):	230.138
Skew:	0.606	Prob(JB):	1.06e-50
Kurtosis:	3.527	Cond. No.	5.43e+07

Notes:

[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

[2] The condition number is large, 5.43e+07. This might indicate that there are

strong multicollinearity or other numerical problems.

Then we create another Regression Model

```
regression_model = LinearRegression()
regression_model.fit(X_train, y_train)
```

```
In [178]: mse = np.mean((regression_model.predict(X_test)-y_test)**2)
```

```
In [179]: # underroot of mean_sq_error is standard deviation i.e. avg variance between predicted and actual

          import math

          math.sqrt(mse)

Out[179]: 610.4833684454707
```

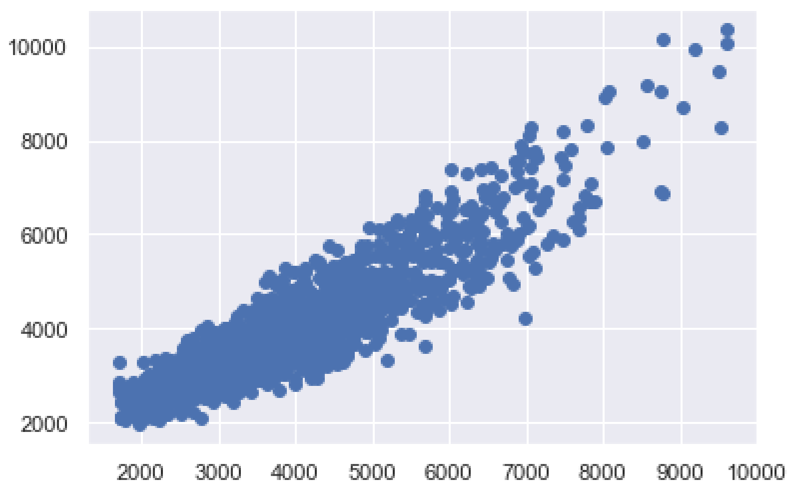
```
In [180]: # Model score - R2 or coeff of determinant
          # R^2=1-RSS / TSS

          regression_model.score(X_test, y_test)

Out[180]: 0.8128013867562254
```

```
In [181]: # predict mileage (mpg) for a set of attributes not in the training or test set
          y_pred = regression_model.predict(X_test)
```

Predict mileage (mpg) for a set of attributes not in the training or test set

Since this is regression, plot the predicted y value vs actual y values for the test data

A good model's prediction will be close to actual leading to high R



Now we get the value of coefficient of determination

```
In [184]: print('The variation in the independent variable which is explained by the dependent variable is',round(model.rsquared*100,4),'%'

          The variation in the independent variable which is explained by the dependent variable is 80.8155 %
```

Then we get the Predictions on test set

```
ypred = model.predict(X_testc)
print(ypred)
```

```
2744    2502.105077
3992    4059.066631
3332    4011.357228
472     3542.932786
1168    3147.634040
           ...
1360    2498.504349
1493    3098.409102
4213    5500.980064
3586    3147.504278
2019    4103.501779
Length: 1356, dtype: float64
```

The Root Mean Square Error

```
In [187]: print("The Root Mean Square Error (RMSE) of the model is for testing set is",np.sqrt(mean_squared_error(y_test,y_pred)))
```

```
The Root Mean Square Error (RMSE) of the model is for testing set is 610.4833684454703
```

```
regression_model = LinearRegression()
regression_model.fit(X_train, y_train)
```

```
LinearRegression()
```

```
print('The coefficient of determination R^2 of the prediction on Train set',regression_model.score(X_train, y_train))
```

```
The coefficient of determination R^2 of the prediction on Train set 0.8081545628073115
```

```
print('The coefficient of determination R^2 of the prediction on Test set',regression_model.score(X_test, y_test))
```

```
The coefficient of determination R^2 of the prediction on Test set 0.8128013867562254
```

```
print("The Root Mean Square Error (RMSE) of the model is for testing set is",np.sqrt(mean_squared_error(y_test,regression_model.p
```

```
The Root Mean Square Error (RMSE) of the model is for testing set is 610.4833684454703
```

Let us explore the coefficients for each of the independent attributes

```
The coefficient for Age is 22.305433275008976
The coefficient for CustTenure is 23.58367512263337
The coefficient for ExistingProdType is 37.012483187760225
The coefficient for NumberOfPolicy is 3.606394132314583
The coefficient for MonthlyIncome is 0.032234148060110934
The coefficient for Complaint is 29.68818504147072
The coefficient for ExistingPolicyTenure is 32.75681083566654
The coefficient for SumAssured is 0.003434757827044166
The coefficient for LastMonthCalls is -1.7463113487215107
The coefficient for CustCareScore is 3.4901993912915645
The coefficient for Channel_Online is 15.234739260186267
The coefficient for Channel_Third_Party_Partner is 19.211451094874846
The coefficient for Occupation_Large_Business is -636.7982076762647
The coefficient for Occupation_Salaried is -674.8400296635879
The coefficient for Occupation_Small_Business is -692.2618592484126
The coefficient for EducationField_Engineer is -22.74738361107367
The coefficient for EducationField_Graduate is -8.038410258729714
The coefficient for EducationField_MBA is -148.57315460426764
The coefficient for EducationField_Post_Graduate is -40.963291955974306
The coefficient for EducationField_Under_Graduate is 0.7167212310898241
The coefficient for Gender_Male is 9.905717143085154
The coefficient for Designation_Exe is -943.0768585468529
The coefficient for Designation_Executive is -482.5650739146594
The coefficient for Designation_Manager is -478.36488278835844
The coefficient for Designation_Senior_Manager is -288.3460989978638
The coefficient for Designation_VP is 112.33525012534695
The coefficient for MaritalStatus_Married is 0.3029246339521265
The coefficient for MaritalStatus_Single is 5.749016483724099
The coefficient for Zone_North is -83.13005166716042
The coefficient for Zone_South is -174.99962343639467
The coefficient for Zone_West is -79.13496496905766
The coefficient for PaymentMethod_Monthly is 149.5088264845823
The coefficient for PaymentMethod_Quarterly is 132.06914121520262
The coefficient for PaymentMethod_Yearly is -58.46534054523914
The coefficient for Agglo_Clusters is 35.208771596169754
```

Now let us check the intercept for the model

In [193]:
```
# Let us check the intercept for the model

intercept = regression_model.intercept_[0]

print("The intercept for our model is {}".format(intercept))
```

```
The intercept for our model is 1302.718206585811
```

In [194]:
```
regression_model.score(X_train, y_train)
```

Out[194]:
```
0.8081545628073115
```

In [195]:
```
regression_model.score(X_test, y_test)
```

Out[195]:
```
0.8128013867562254
```

```
data_train = pd.concat([X_train, y_train], axis=1)
data_train.head()
```

	Age	CustTenure	ExistingProdType	NumberOfPolicy	MonthlyIncome	Complaint	ExistingPolicyTenure	SumAssured	LastMonthCalls	CustCareScore	Cha
4089	21.0	11.0	6	4.0	22165.0	0	1.0	663177.0	5	3.0	
696	13.0	6.0	4	4.0	17743.0	0	1.0	408799.0	2	5.0	
171	18.0	22.0	3	3.0	20296.0	0	1.0	617404.0	7	5.0	
102	3.0	13.0	3	3.0	18161.0	1	3.0	581152.0	0	5.0	
243	25.0	10.0	4	5.0	25266.0	0	6.0	717554.0	11	2.0	

In [197]:
```
regression_model_scaled = LinearRegression()
regression_model_scaled.fit(X_train_scaled, y_train_scaled)
```

Out[197]:
```
LinearRegression()
```

Let us explore the coefficients for each of the independent attributes

```
The coefficient for Age is 0.13767054291276307
The coefficient for CustTenure is 0.14673231545576854
The coefficient for ExistingProdType is 0.026583240287272193
The coefficient for NumberOfPolicy is 0.0037266361191006896
The coefficient for MonthlyIncome is 0.11051475089919886
The coefficient for Complaint is 0.009653017910747817
The coefficient for ExistingPolicyTenure is 0.07721280682869969
The coefficient for SumAssured is 0.5911963397769338
The coefficient for LastMonthCalls is -0.0044820220773235525
The coefficient for CustCareScore is 0.0034218816627441077
The coefficient for Channel_Online is 0.003357526778051917
The coefficient for Channel_Third_Party_Partner is 0.005414297785330311
The coefficient for Occupation_Large_Business is -0.1289981397245527
The coefficient for Occupation_Salaried is -0.24104793809459416
The coefficient for Occupation_Small_Business is -0.24364887911792552
The coefficient for EducationField_Engineer is -0.004600537950311366
The coefficient for EducationField_Graduate is -0.0028361177496419795
The coefficient for EducationField_MBA is -0.01323705141140738
The coefficient for EducationField_Post_Graduate is -0.006969724212726316
The coefficient for EducationField_Under_Graduate is 0.00023675340244105544
The coefficient for Gender_Male is 0.0034762734938712687
The coefficient for Designation_Exe is -0.10955626710497682
The coefficient for Designation_Executive is -0.16382497709257654
The coefficient for Designation_Manager is -0.163620298237565
The coefficient for Designation_Senior_Manager is -0.07267652783677697
The coefficient for Designation_VP is 0.017791198805611586
The coefficient for MaritalStatus_Married is 0.0001082040270019586
The coefficient for MaritalStatus_Single is 0.0019141209599569647
The coefficient for Zone_North is -0.029258500434660367
The coefficient for Zone_South is -0.0038478363594147275
The coefficient for Zone_West is -0.027982323754654932
The coefficient for PaymentMethod_Monthly is 0.028007029851631345
The coefficient for PaymentMethod_Quarterly is 0.011995805747694384
The coefficient for PaymentMethod_Yearly is -0.01947101649539555
The coefficient for Agglo_Clusters is 0.029369659107792462
```

```python
intercept = regression_model_scaled.intercept_[0]

print("The intercept for our model is {}".format(intercept))
```

```
The intercept for our model is 7.798336340434967e-18
```

```python
# Model score - R2 or coeff of determinant
# R^2=1-RSS / TSS

regression_model_scaled.score(X_test_scaled, y_test_scaled)
```

```
0.8125601603968964
```

```python
# Let us check the sum of squared errors by predicting value of y for training cases and
# subtracting from the actual y for the training cases

mse_scaled = np.mean((regression_model_scaled.predict(X_test_scaled)-y_test_scaled)**2)
```

```python
# underroot of mean_sq_error is standard deviation i.e. avg variance between predicted and actual

import math

math.sqrt(mse_scaled)
```

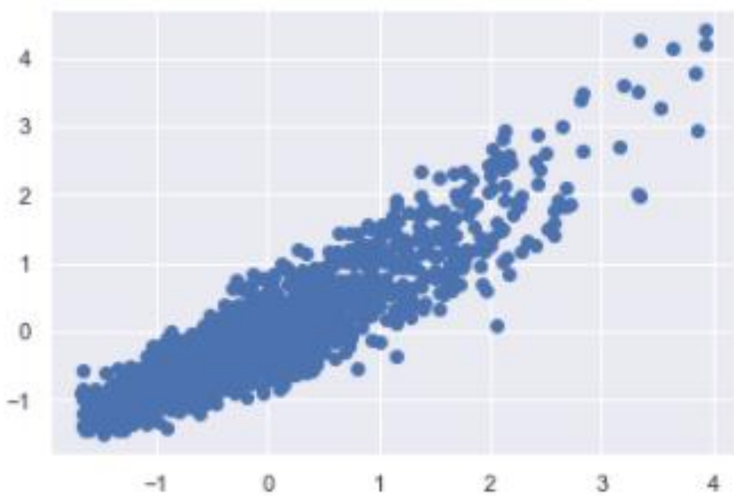
```
0.432943229076404
```

```python
# predict mileage (mpg) for a set of attributes not in the training or test set
y_pred_scaled = regression_model_scaled.predict(X_test_scaled)
```

Predict mileage (mpg) for a set of attributes not in the training or test set

Since this is regression, plot the predicted y value vs actual y values for the test data. A good model's prediction will be close to actual leading to high R and R2 values

```
<matplotlib.collections.PathCollection at 0x222969876d0>
```



We again check the variance inflation factor

```
Age ---> 4.974204076010754
CustTenure ---> 5.0042780103168765
ExistingProdType ---> 64.01833054809883
NumberOfPolicy ---> 7.8321774219042455
MonthlyIncome ---> 141.82458279669777
Complaint ---> 1.414558834499288
ExistingPolicyTenure ---> 2.8678193653545065
SumAssured ---> 13.290246233456738
LastMonthCalls ---> 3.1858820162173864
CustCareScore ---> 6.091559455568143
Channel_Online ---> 1.1686497715165352
```

Let’s invoke the LinearRegression function and find the best fit model on training data.

```
regression_model_out = LinearRegression()
regression_model_out.fit(X_train_no_out, y_train_no_out)
```

```
LinearRegression()
```

Now let us explore the coefficients for each of the independent attributes

```
The coefficient for Age is 21.16035136703995
The coefficient for CustTenure is 22.43803112721254
The coefficient for ExistingProdType is 35.64162751908745
The coefficient for NumberOfPolicy is 4.385210311006258
The coefficient for MonthlyIncome is 0.03218690988832312
The coefficient for Complaint is 25.336989517761737
The coefficient for ExistingPolicyTenure is 35.68936973913441
The coefficient for SumAssured is 0.0035582576401367305
The coefficient for LastMonthCalls is -0.4588578496119094
The coefficient for CustCareScore is 2.40072559100181
The coefficient for Channel_Online is 8.067521549077147
The coefficient for Channel_Third Party Partner is 17.41914062086986
The coefficient for Occupation_Large Business is -624.9404677783616
The coefficient for Occupation_Salaried is -667.5143860777937
The coefficient for Occupation_Small Business is -693.2541275356853
The coefficient for EducationField_Engineer is -38.048640773593306
The coefficient for EducationField_Graduate is -20.382776609002768
The coefficient for EducationField_MBA is -176.00489806226838
The coefficient for EducationField_Post Graduate is -49.2307803636899
The coefficient for EducationField_Under Graduate is 6.275630855759628
The coefficient for Gender_Male is 12.212745365363402
The coefficient for Designation_Exe is -951.6472580695299
The coefficient for Designation_Executive is -479.2809087248839
The coefficient for Designation_Manager is -478.8052227116883
The coefficient for Designation_Senior Manager is -286.382061263671
The coefficient for Designation_VP is 57.25448765074445
The coefficient for MaritalStatus_Married is -7.7643848844325065
The coefficient for MaritalStatus_Single is -0.27209760432839497
The coefficient for Zone_North is -83.64401292733379
The coefficient for Zone_South is -195.50942258990926
The coefficient for Zone_West is -88.35484503904541
The coefficient for PaymentMethod_Monthly is 124.95660663201004
The coefficient for PaymentMethod_Quarterly is 143.22889619995138
The coefficient for PaymentMethod_Yearly is -52.89556145000119
```

```python
intercept = regression_model_out.intercept_[0]

print("The intercept for our model is {}".format(intercept))
```

```
The intercept for our model is 1412.8814770774493
```

```python
# Model score - R2 or coeff of determinant
# R^2=1-RSS / TSS

regression_model_out.score(X_test_no_out, y_test_no_out)
```

```
0.8039846439306599
```

	variables	VIF
30	Zone_South	1.097750
33	PaymentMethod_Quarterly	1.124055
11	Channel_Online	1.168744
12	Channel_Third_Party_Partner	1.285194
6	Complaint	1.415376
22	Designation_Exe	2.129217
18	EducationField_MBA	2.250673
26	Designation_VP	2.364060
32	PaymentMethod_Monthly	2.491524
21	Gender_Male	2.535430
28	MaritalStatus_Single	2.871670
7	ExistingPolicyTenure	2.960278
9	LastMonthCalls	3.186624
34	PaymentMethod_Yearly	3.246193
27	MaritalStatus_Married	3.895606
20	EducationField_Under_Graduate	3.997850
25	Designation_Senior_Manager	4.654740
19	EducationField_Post_Graduate	5.132235
1	Age	5.384959
2	CustTenure	5.448530
10	CustCareScore	6.092959
4	NumberOfPolicy	7.832282
24	Designation_Manager	11.160432
23	Designation_Executive	12.938618
16	EducationField_Engineer	20.769377
8	SumAssured	27.123217
29	Zone_North	30.124808
17	EducationField_Graduate	31.052427
31	Zone_West	40.643026
13	Occupation_Large_Business	45.003817
35	Agglo_Clusters	49.216890
0	AgentBonus	49.676191
3	ExistingProdType	64.107943
15	Occupation_Small_Business	121.653120
5	MonthlyIncome	143.287388
14	Occupation_Salaried	163.510600

PROBLEM 1.C

Interpretation of the model(s)

Resolution:

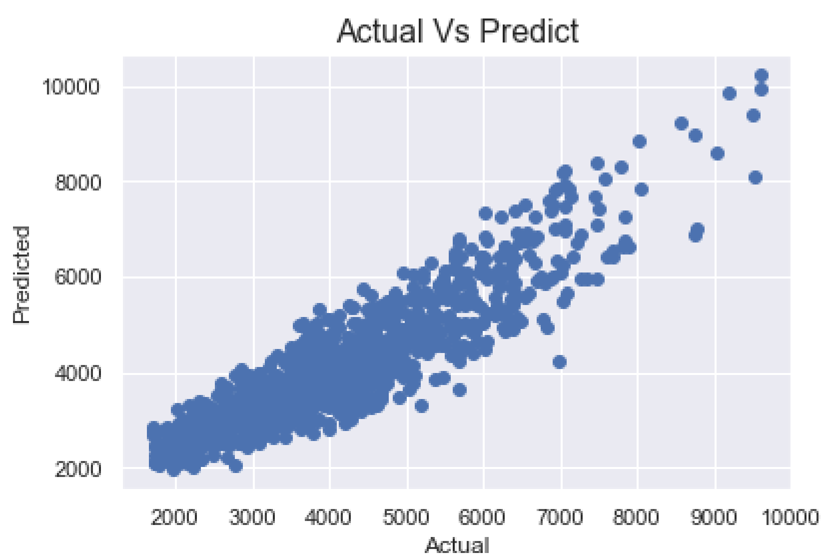
The final Linear Regression equation is

AgentBonus = b0 * Intercept + b1 * Zone_South + b2 * PaymentMethod_Quarterly + b3 * Channel_Online + b4 * Channel_Third_Party_Partner + b5 * Complaint + b6 * Designation_Exe + b7 * Designation_VP + b8 * EducationField_MBA + b9 * PaymentMethod_Monthly + b10 * Gender_Male + b11 * MaritalStatus_Single + b12 * ExistingPolicyTenure + b13 * LastMonthCalls + b14 * PaymentMethod_Yearly + b15 * Designation_Senior_Manager + b16 * MaritalStatus_Married + b17 * EducationField_Under_Graduate + b18 * EducationField_Post_Graduate + b19 * Age + b20 * CustTenure + b21 * CustCareScore + b22 * NumberOfPolicy + b23 * Designation_Manager + b24 * Designation_Executive + b25 * EducationField_Engineer + b26 * SumAssured + b27 * Zone_North + b28 * EducationField_Graduate + b29 * Occupation_Large_Business + b30 * Zone_West + b31 * ExistingProdType + b32 * MonthlyIncome + b33 * Occupation_Small_Business

AgentBonus = (843.69) * Intercept + (-0.74) * Zone_South + (121.99) * PaymentMethod_Quarterly + (16.45) * Channel_Online + (8.64) * Channel_Third_Party_Partner + (31.43) * Complaint + (-897.0) * Designation_Exe + (19.69) * Designation_VP + (-134.73) * EducationField_MBA + (179.77) * PaymentMethod_Monthly + (3.74) * Gender_Male + (13.12) * MaritalStatus_Single + (35.55) * ExistingPolicyTenure + (-2.62) * LastMonthCalls + (-71.13) * PaymentMethod_Yearly + (-261.65) * Designation_Senior_Manager + (-16.47) * MaritalStatus_Married + (15.54) * EducationField_Under_Graduate + (-76.12) * EducationField_Post_Graduate + (23.04) * Age + (24.0) * CustTenure + (6.01) * CustCareScore + (1.52) * NumberOfPolicy + (-447.69) * Designation_Manager + (-469.53) * Designation_Executive + (-39.86) * EducationField_Engineer + (0.0) * SumAssured + (-44.05) * Zone_North + (-63.73) * EducationField_Graduate + (-39.31) * Occupation_Large_Business + (-44.97) * Zone_West + (54.11) * ExistingProdType + (0.03) * MonthlyIncome + (-92.01) * Occupation_Small_Business

When Age increases by 1 unit, AgentBonus increases by 23.04 units, keeping all other predictors constant. similarly, when MonthlyIncome increases by 1 unit, AgentBonus increases by 0.03 units, keeping all other predictors constant.

There are also some negative co-efficient values. Occupation_Large_Business has its corresponding co-efficient as -39.31. This implies, when the Occupation is Large business, the AgentBonus decreases by 39.31 units, keeping all other predictors constant.



Problem 2: Model Tuning and business implication

PROBLEM 2.A

Ensemble modelling (if necessary)

Resolution:

We are scaling the data for ANN. Without scaling it will give very poor results. Computations becomes easier.

	Train RMSE	Test RMSE	Training Score	Test Score
Linear Regression	611.612103	614.934255	0.808172	0.813200
Decision Tree Regressor	0.000000	744.328618	1.000000	0.726316
Random Forest Regressor	200.059407	519.298515	0.979475	0.866785
ANN Regressor	4315.881788	4303.418081	0.969355	0.781622

In [244]:
```
param_grid = {
    'max_depth': [10,15,20,25,30],
    'min_samples_leaf': [3, 15,30],
    'min_samples_split': [15,30,35,40,50],
}

dtr=tree.DecisionTreeRegressor(random_state=8)

grid_search = GridSearchCV(estimator = dtr, param_grid = param_grid, cv = 3)


grid_search.fit(X_train_tun,y_train_tun)

print(grid_search.best_params_)
```

```
{'max_depth': 10, 'min_samples_leaf': 3, 'min_samples_split': 40}
```

In [245]:
```
param_grid = {
    'max_depth': [7,10],
    'max_features': [4, 6],
    'min_samples_leaf': [3, 15,30],
    'min_samples_split': [30, 40,100],
    'n_estimators': [300, 500]
}

rfr = RandomForestRegressor(random_state=8)

grid_search = GridSearchCV(estimator = rfr, param_grid = param_grid, cv = 3)
```

grid_search.fit(X_train_tun,y_train_tun)

```
#best_params_rfr={'max_depth': 10, 'max_features': 6, 'min_samples_leaf': 3, 'min_samples_split': 30, 'n_estimators': 500}
```

Using Grid Search for ANN

```python
param_grid = {
    'hidden_layer_sizes':[(500),(100,100)],
    # keeping these simple because it would take too much time to run on low-end computers
    "activation": ["tanh", "relu"],
    "solver": ["sgd", "adam"]}

annr = MLPRegressor(max_iter=1000, random_state=8)

grid_search = GridSearchCV(estimator = annr, param_grid = param_grid, cv = 3)
```

```
Out[250]: GridSearchCV(cv=3, estimator=MLPRegressor(max_iter=1000, random_state=8),
                       param_grid={'activation': ['tanh', 'relu'],
                                   'hidden_layer_sizes': [500, (100, 100)],
                                   'solver': ['sgd', 'adam']})
```

```python
print(grid_search.best_params_)
```

```
{'activation': 'relu', 'hidden_layer_sizes': 500, 'solver': 'sgd'}
```

best_params_annr={'activation': 'relu', 'hidden_layer_sizes': 500, 'solver': 'sgd'}

	Train RMSE	Test RMSE	Training Score	Test Score
Linear Regression	611.612103	614.934255	0.808172	0.813200
Decision Tree Regressor	508.120305	597.920356	0.867599	0.823394
Random Forest Regressor	542.802536	620.584032	0.848908	0.809752
ANN Regressor	0.369968	0.424350	0.863124	0.819927

Without tuning

	Train RMSE	Test RMSE	Training Score	Test Score
Linear Regression	613.101894	610.483368	0.808155	0.812801
Decision Tree Regressor	513.260619	593.156926	0.865550	0.823277
Random Forest Regressor	552.781054	625.002852	0.844047	0.803791
ANN Regressor	0.369968	0.424350	0.863124	0.819927

Final Output

	Train RMSE	Test RMSE	Training Score	Test Score
Linear Regression	0.438002	0.432943	0.808155	0.812560
Decision Tree Regressor	0.366675	0.428688	0.865550	0.816227
Random Forest Regressor	0.394837	0.445444	0.844103	0.801580
ANN Regressor	0.369968	0.424350	0.863124	0.819927

PROBLEM 2.B

Any other model tuning measures (if applicable)

Resolution:

Tuning

In [217]:
```
df_vif=df[['Zone_South','PaymentMethod_Quarterly','Channel_Online','Channel_Third_Party_Partner'
   ,'Complaint','Designation_Exe','Designation_VP','EducationField_MBA','PaymentMethod_Monthly'
   ,'Gender_Male','MaritalStatus_Single','ExistingPolicyTenure','LastMonthCalls','PaymentMethod_Yearly'
   ,'Designation_Senior_Manager','MaritalStatus_Married','EducationField_Under_Graduate','AgentBonus','EducationField_Post_Gradua
   ,'Age','CustTenure','CustCareScore','NumberOfPolicy','Designation_Manager','Designation_Executive','EducationField_Engineer'
    ,'SumAssured','Zone_North','EducationField_Graduate','Occupation_Large_Business','Zone_West','ExistingProdType','MonthlyIncom
          ,'Occupation_Small_Business']]
#score is max when we remove Occupation_Salaried
```

In [218]:
```
df_vif
```

Out[218]:

	Zone_South	PaymentMethod_Quarterly	Channel_Online	Channel_Third_Party_Partner	Complaint	Designation_Exe	Designation_VP	EducationField_MBA
0	0	0	0	0	1	0	0	0
1	0	0	0	1	0	0	0	0
2	0	0	0	0	1	1	0	0
3	0	0	0	1	1	0	0	0
4	0	0	0	0	0	0	0	0
...	...	...	...	...	...	...	...	...
4515	0	0	0	0	0	0	0	0
4516	0	0	0	0	0	0	0	0
4517	0	0	0	0	0	0	0	0
4518	0	0	1	0	0	0	0	0
4519	0	0	0	0	0	0	0	0

4520 rows × 34 columns

We now invoke the LinearRegression function and find the best fit model on training data

```
regression_model_vif = LinearRegression()
regression_model_vif.fit(X_train_vif, y_train_vif)
```

```
LinearRegression()
```

Let us explore the coefficients for each of the independent attributes

```
The coefficient for Zone_South is -177.53074739826755
The coefficient for PaymentMethod_Quarterly is 140.20712725616963
The coefficient for Channel_Online is 13.012341753675729
The coefficient for Channel_Third_Party_Partner is 20.487115418348655
The coefficient for Complaint is 31.685361413976963
The coefficient for Designation_Exe is -914.2840345317121
The coefficient for Designation_VP is 83.61466592603371
The coefficient for EducationField_MBA is -152.46678008724237
The coefficient for PaymentMethod_Monthly is 161.46517974037127
The coefficient for Gender_Male is 10.083903510394341
The coefficient for MaritalStatus_Single is 7.317067441019994
The coefficient for ExistingPolicyTenure is 32.914820300413645
The coefficient for LastMonthCalls is -1.5066627383331075
The coefficient for PaymentMethod_Yearly is -63.09658265704684
The coefficient for Designation_Senior_Manager is -234.90779982050515
The coefficient for MaritalStatus_Married is 0.2273700794131594
The coefficient for EducationField_Under_Graduate is -1.8847983945686386
The coefficient for EducationField_Post_Graduate is -37.79407017070104
The coefficient for Age is 22.158517692104205
The coefficient for CustTenure is 23.395954345736282
The coefficient for CustCareScore is 3.3671519083631627
The coefficient for NumberOfPolicy is 3.8656550571600334
The coefficient for Designation_Manager is -436.56179719317475
The coefficient for Designation_Executive is -452.4643380524982
The coefficient for EducationField_Engineer is -17.317260792697528
The coefficient for SumAssured is 0.003433789096447981
The coefficient for Zone_North is -79.65862392165423
The coefficient for EducationField_Graduate is -11.888898369021842
The coefficient for Occupation_Large_Business is 29.913578690562634
The coefficient for Zone_West is -78.22206217344456
The coefficient for ExistingProdType is 43.98633586581726
The coefficient for MonthlyIncome is 0.02745411921963592
The coefficient for Occupation_Small_Business is -19.367644879940702
The intercept for our model is 810.7915091307291
```

Model score - R2 or coeff of determinant

R^2=1–RSS / TSS

```
0.8131119381670872
```

We can see that the scaled output has a better score.

Now we see stats model formula

```
Out[226]: Intercept                          843.694069
          Zone_South                          -0.743471
          PaymentMethod_Quarterly            121.985130
          Channel_Online                      16.452712
          Channel_Third_Party_Partner          8.639260
          Complaint                           31.425465
          Designation_Exe                   -896.998779
          Designation_VP                      19.689241
          EducationField_MBA                -134.730077
          PaymentMethod_Monthly              179.770948
          Gender_Male                          3.736511
          MaritalStatus_Single                13.121798
          ExistingPolicyTenure                35.545569
          LastMonthCalls                      -2.622818
          PaymentMethod_Yearly               -71.133166
          Designation_Senior_Manager        -261.650782
          MaritalStatus_Married              -16.469356
          EducationField_Under_Graduate       15.535704
          EducationField_Post_Graduate       -76.117914
          Age                                 23.035100
          CustTenure                          24.001376
          CustCareScore                        6.006542
          NumberOfPolicy                       1.516024
          Designation_Manager               -447.692353
          Designation_Executive             -469.527655
          EducationField_Engineer            -39.861369
          SumAssured                           0.003425
          Zone_North                         -44.049140
          EducationField_Graduate            -63.733576
          Occupation_Large_Business          -39.310314
          Zone_West                          -44.965068
          ExistingProdType                    54.113346
          MonthlyIncome                        0.025356
          Occupation_Small_Business          -92.005135
          dtype: float64
```

lm1.summary

```
                            OLS Regression Results
==============================================================================
Dep. Variable:             AgentBonus   R-squared:                       0.810
Model:                            OLS   Adj. R-squared:                  0.809
Method:                 Least Squares   F-statistic:                     579.7
Date:                Sun, 16 Jan 2022   Prob (F-statistic):               0.00
Time:                        13:13:01   Log-Likelihood:                -35414.
No. Observations:                4520   AIC:                         7.090e+04
Df Residuals:                    4486   BIC:                         7.111e+04
Df Model:                          33
Covariance Type:            nonrobust
-----------------------------------------------------------------------------------------------
```

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	843.6941	158.479	5.324	0.000	532.996	1154.392
Zone_South	-0.7435	262.924	-0.003	0.998	-516.204	514.718
PaymentMethod_Quarterly	121.9851	74.532	1.637	0.102	-24.135	268.105
Channel_Online	16.4527	30.665	0.537	0.592	-43.665	76.571
Channel_Third_Party_Partner	8.6393	23.756	0.364	0.716	-37.935	55.213
Complaint	31.4255	20.270	1.550	0.121	-8.314	71.165
Designation_Exe	-896.9988	78.666	-11.403	0.000	-1051.223	-742.775
Designation_VP	19.6892	61.233	0.322	0.748	-100.358	139.736
EducationField_MBA	-134.7301	107.913	-1.249	0.212	-346.292	76.832
PaymentMethod_Monthly	179.7709	51.123	3.516	0.000	79.544	279.997
Gender_Male	3.7365	18.855	0.198	0.843	-33.229	40.702
MaritalStatus_Single	13.1218	27.369	0.479	0.632	-40.534	66.778
ExistingPolicyTenure	35.5456	2.949	12.054	0.000	29.764	41.327
LastMonthCalls	-2.6228	2.771	-0.946	0.344	-8.056	2.810
PaymentMethod_Yearly	-71.1332	29.098	-2.445	0.015	-128.179	-14.087
Designation_Senior_Manager	-261.6508	43.913	-5.958	0.000	-347.742	-175.559
MaritalStatus_Married	-16.4694	25.455	-0.647	0.518	-66.374	33.435
EducationField_Under_Graduate	15.5357	32.617	0.476	0.634	-48.410	79.481
EducationField_Post_Graduate	-76.1179	89.154	-0.854	0.393	-250.903	98.668
Age	23.0351	1.204	19.134	0.000	20.675	25.395
CustTenure	24.0014	1.208	19.865	0.000	21.633	26.370
CustCareScore	6.0065	6.712	0.895	0.371	-7.152	19.165
NumberOfPolicy	1.5160	6.633	0.229	0.819	-11.488	14.520
Designation_Manager	-447.6924	46.393	-9.650	0.000	-538.646	-356.739
Designation_Executive	-469.5277	54.362	-8.637	0.000	-576.104	-362.952
EducationField_Engineer	-39.8614	138.810	-0.287	0.774	-311.997	232.275
SumAssured	0.0034	5.01e-05	68.353	0.000	0.003	0.004
Zone_North	-44.0491	78.644	-0.560	0.575	-198.229	110.131
EducationField_Graduate	-63.7336	80.236	-0.794	0.427	-221.036	93.569
Occupation_Large_Business	-39.3103	129.498	-0.304	0.761	-293.190	214.570
Zone_West	-44.9651	78.229	-0.575	0.565	-198.333	108.402
ExistingProdType	54.1133	18.847	2.871	0.004	17.165	91.062
MonthlyIncome	0.0254	0.004	5.971	0.000	0.017	0.034
Occupation_Small_Business	-92.0051	75.805	-1.214	0.225	-240.621	56.611

Omnibus:	261.417	Durbin-Watson:	2.018
Prob(Omnibus):	0.000	Jarque-Bera (JB):	311.623
Skew:	0.599	Prob(JB):	2.15e-68
Kurtosis:	3.467	Cond. No.	1.95e+07

Notes:

[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

[2] The condition number is large, 1.95e+07. This might indicate that there are

strong multicollinearity or other numerical problems.

```
for i,j in np.array(lm1.params.reset_index()):
    print('({}) * {} +'.format(round(j,2),i),end=' ')
```

```
(843.69) * Intercept + (-0.74) * Zone_South + (121.99) * PaymentMethod_Quarterly + (16.45) * Channel_Online + (8.64) * Channel_
Third_Party_Partner + (31.43) * Complaint + (-897.0) * Designation_Exe + (19.69) * Designation_VP + (-134.73) * EducationField_
MBA + (179.77) * PaymentMethod_Monthly + (3.74) * Gender_Male + (13.12) * MaritalStatus_Single + (35.55) * ExistingPolicyTenure
+ (-2.62) * LastMonthCalls + (-71.13) * PaymentMethod_Yearly + (-261.65) * Designation_Senior_Manager + (-16.47) * MaritalStatu
s_Married + (15.54) * EducationField_Under_Graduate + (-76.12) * EducationField_Post_Graduate + (23.04) * Age + (24.0) * CustTe
nure + (6.01) * CustCareScore + (1.52) * NumberOfPolicy + (-447.69) * Designation_Manager + (-469.53) * Designation_Executive +
(-39.86) * EducationField_Engineer + (0.0) * SumAssured + (-44.05) * Zone_North + (-63.73) * EducationField_Graduate + (-39.31)
* Occupation_Large_Business + (-44.97) * Zone_West + (54.11) * ExistingProdType + (0.03) * MonthlyIncome + (-92.01) * Occupatio
n_Small_Business +
```

```
X_train_tun, X_test_tun, y_train_tun, y_test_tun = train_test_split(X_vif, Y, test_size=0.25, random_state=8)
X_train_scaled_tun  = X_train.apply(zscore)
X_test_scaled_tun = X_test.apply(zscore)
y_train_scaled_tun = y_train.apply(zscore)
y_test_scaled_tun = y_test.apply(zscore)
```

```
regression_model_tun = LinearRegression()
regression_model_tun.fit(X_train_tun, y_train_tun)
regression_model_tun.score(X_test_tun, y_test_tun)
```

```
0.8132000652084972
```

```
regression_model_scaled_tun = LinearRegression()
regression_model_scaled_tun.fit(X_train_scaled_tun, y_train_scaled_tun)
regression_model_scaled_tun.score(X_test_scaled_tun, y_test_scaled_tun)
```

```
0.8125601603968964
```

The overall P value is less than alpha, so rejecting H0 and accepting Ha that at least 1 regression co-efficient is not 0. Here all regression co-efficients are not 0

PROBLEM 2.C

Interpretation of the most optimum model and its implication on the business

Resolution:

When Age increases by 1 unit, AgentBonus increases by 23.04 units, keeping all other predictors constant. similarly, when MonthlyIncome increases by 1 unit, AgentBonus increases by 0.03 units, keeping all other predictors constant.

There are also some negative co-efficient values. Occupation_Large_Business has its corresponding co-efficient as -39.31. This implies, when the Occupation is large business, the AgentBonus decreases by 39.31 units, keeping all other predictors constant.

The End

Thakur Arun Singh

******************************^^^**************************