Understanding Data Coursework

Shekhar Jogawade

210268199

Abstract

Most of the companies in the industry are dealing with customer churn rate. Customer churn has a significant impact on your business as it lowers revenues and profits. Yet surprisingly, more than 2 out of 3 companies have no strategy for preventing customer churn. Customer churn is calculated by the number of customers who leave your company during a given time period. In a more down-to-earth sense, churn rate shows how your business is doing with keeping customers by your side. By accurately predicting churn rate we will be helping Lu's communication. Lu's communications would like to use a targeted approach to identify in advance customers who are likely to churn, which can be then targeted with special programs or incentives. This approach can bring in a huge loss if churn predictions are inaccurate, as money would be wasted to incentivise customers who would have stayed anyway.

This prediction task of churn customer was done based on different features with the help of Machine Learning algorithms such as Logistic Regression, K-Nearest Neighbor, Random Forest, Support Vector Machine and Naïve Bayes. We have also used Different evaluation metrics was used namely Accuracy Score, Precision, Recall, F1 Score, Confusion Matrix and Log loss.

2 Introduction

2.1 Importance

Customer churn is calculated by the number of customers who leave your company during a given time period. In a more down-to-earth sense, churn rate shows how your business is doing with keeping customers by your side.But why does churn matter so much for businesses in the first place? Well, the short answer is – because it costs ***too much*** for customers to stop doing business with you. Companies spends 5 Times more money to attract new customer than to provide service to existing customer. It will cost Lu’s Communication **16 times more to bring a new customer up to the same level** as an existing customer.The second reason lies in the fact that the more customers a business retains, the more revenue it makes!

According to the study of Harvard Business School, on average **a 5% increase in customer retention** rates results in **25% – 95% increase of profits.** And the lion’s share – 65% of a company’s business comes from existing customers!. Similarly KPMG believes that customer retention contributes to 52% as most significant retail revenue driver. Therefore Accurately predicting the churn rate of Lu’s Communication is a critical task.

2.2 Objective

The main purpose of doing this exercise, predicting Accurate Churn rate based on investigating on Different features of customers like gender, location and dependents etc. by using Machine Learning approach. The prediction task was done for predicting the churn rate of customer that Lu’s Communication Might lose in the future.. Therefore, in this project different machine learning algorithms such as, Logistic Regression, K-Nearest Neighbor, Support Vector Machine and Naïve Bayes were used to predict whether the customer is going to churn or not. For evaluating the models, 5 metrics was used such as Accuracy Score, Precision, Recall, F1 Score, Confusion Matrix and Log loss.

2.3 Methodology

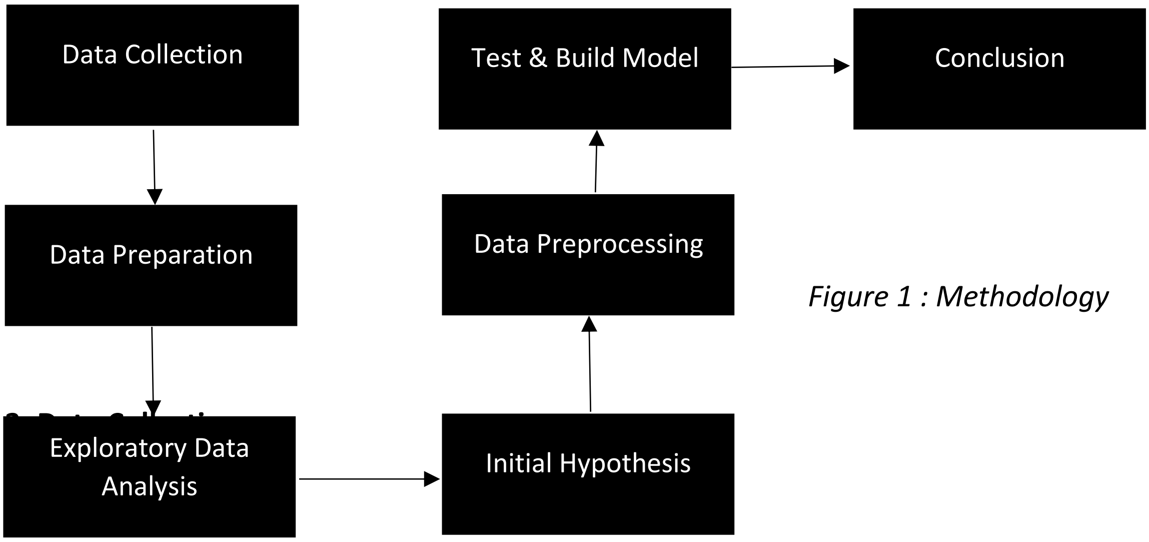
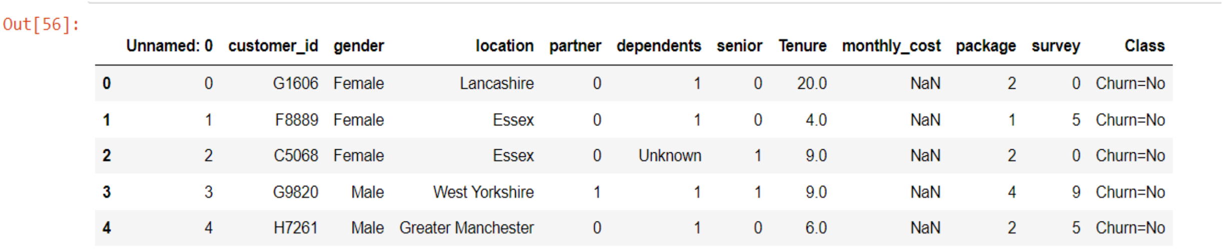


Figure 1 : Methodology

Lu’s communication has already provided the data (data.csv). However, this data is not perfect.

It may contain outliers, missing values and irrelevant (not related to churn) information.

3.1 Dataset :



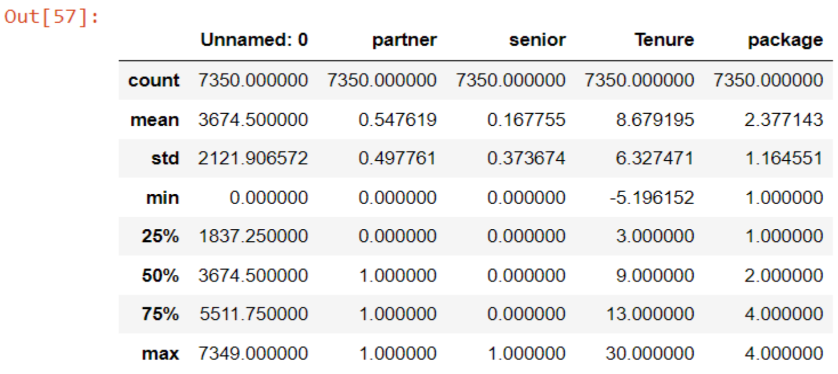
Out[56]:

	Unnamed: 0	customer_id	gender	location	partner	dependents	senior	Tenure	monthly_cost	package	survey	Class
0	0	G1606	Female	Lancashire	0	1	0	20.0	NaN	2	0	Churn=No
1	1	F8889	Female	Essex	0	1	0	4.0	NaN	1	5	Churn=No
2	2	C5068	Female	Essex	0	Unknown	1	9.0	NaN	2	0	Churn=No
3	3	G9820	Male	West Yorkshire	1	1	1	9.0	NaN	4	9	Churn=No
4	4	H7261	Male	Greater Manchester	0	1	0	6.0	NaN	2	5	Churn=No

3.2 Feature Information :

Dataset contains 7350 rows and 12 columns. The features of the dataset are:

customer_id: Every customer is given a unique ID
gender: Whether the customer is a male or a female location: Location of the customer
partner: Whether the customer has a partner or not (1=Yes, 0=No)
dependents: Whether the customer has dependents or not (1=Yes, 0=No)
senior: Whether the customer is a senior citizen or not (1=Yes, 0=No)
Tenure: Number of years the customer has stayed with the company. Lu’s communication offers a loyalty scheme; this scheme gives 2% discount on the monthly cost for each year (up to 25 years) the customer remains with the company
monthly_cost: The amount charged to the customer monthly
package: Packages Lu’s communication offers. See table below for more information survey: Score given by customers on the customer service (0=``Poor”, 10=``Excellent”)
Class: Whether the customer churned or not
Basic information about the dataset like minimum value, maximum value, count, 25%, 50%, 75% , mean and standard deviation of the dataset.



Out[57]:

	Unnamed: 0	partner	senior	Tenure	package
count	7350.000000	7350.000000	7350.000000	7350.000000	7350.000000
mean	3674.500000	0.547619	0.167755	8.679195	2.377143
std	2121.906572	0.497761	0.373674	6.327471	1.164551
min	0.000000	0.000000	0.000000	-5.196152	1.000000
25%	1837.250000	0.000000	0.000000	3.000000	1.000000
50%	3674.500000	1.000000	0.000000	9.000000	2.000000
75%	5511.750000	1.000000	0.000000	13.000000	4.000000
max	7349.000000	1.000000	1.000000	30.000000	4.000000

4.Data Preparation :

Data preparation is the process of gathering, combining, structuring and organizing data so it can be used in business intelligence (BI), analytics and data visualization applications. The components of data preparation include data preprocessing, profiling, cleansing, validation and transformation. (SearchBusinessAnalytics, n.d.)

We have implemented the code for understanding the datatype and missing values in given feature.

1. **Customer_id :**
   It is a unique value so every customer will have a unique values assign to them. By default it is a object datatype but we have converted this datatype to string for further ease of manipulation. There are 0 missing values in customer id.

2. **Gender :**
   There are two types of gender present in the dataset. i.e male and female. We converted the misinterpreted datatype from object to string. Gender does not contain any missing values.
   We have also did count and we have seen that there are more female customers than male customers.
   Female Customers : 3689
   Male Customers : 3661

3. **Location :**
   Location contains different regions of United Kingdom. We converted the misinterpreted datatype from object to string. There are 0 missing values in location.

4. **Partner :**
   We converted the misinterpreted datatype from object to string. There are 0 missing values in location. If customers have partner then its 1 otherwise its 0.

5. **Dependents :**
   Datatype of dependents column is string but we can convert it to integer or float. We decided to go for float datatype to avoid any programming errors. If customers have any dependents then its 1 otherwise its 0. But, there are 2208 unknown values, so we decided to convert this unknown values to nan and then in the later phase we have used KNN imputer to impute the values.

6. **Senior :**
   If customer have any senior member then its 1 otherwise its 0. Datatype of this feature is integer therefore we decided not to change it and keep it as it is.

7. **Tenure :**
   Tenure refers to the no of years customer is using service of Lu's Communication. Tenure has float datatype so we decided not to change it. There are negative tenure values. No of years cannot be negative. Therefore we had following options to deal with negative tenure.
   - Change all values < 0 to 0.
   - Take the absolute value i.e. the value `-2` will now be `2`.
   - Remove the columns.

Therefore we decided to use absolute values. We will convert the negative years to positive one.

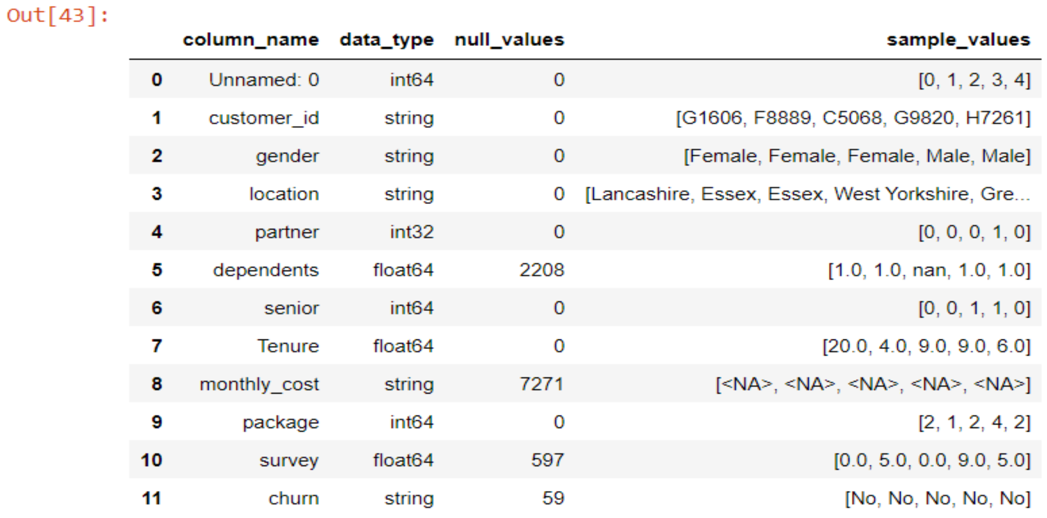
8. **Monthly_cost :**
   This feature has most null values. we'll analyse package value to get monthly value and also we'll calculate tenure in months and we'll also calculate the final monthly value after giving 2% yearly discount and store it in monthly_cost.

9. **Package :**
   Package contains integer values like 1,2,3,4. Package is having integer datatype so we decided not to change the datatype. Package does not have any missing values. There are different package prices are there but package 4 value is missing so we decided to take package 4 values as 44 pounds.

10. **Survey:**
    Survey contains customer score from 0 to 10 to Lu's communication according to their service. It has integer datatype but we decided to change it to float. There are 597 records which does not have any value associated with it. Maybe customers have not responded to the survey.
    We decided to convert this values to nan and then we will use KNN imputer to impute this values in later phase.

11. **Class :**
    We have renamed feature name from "Class" to "churn". There are 59 missing values in this dependent feature. We have converted the datatype from integer to string.
    There are 19 different values then yes and no i.e. "Y$e$s$$". We can see that it looks like it should be Yes. Therefore we have converted this values in "Yes".

4.1 Summary of Data Preparation :



Out[43]:

	column_name	data_type	null_values	sample_values
0	Unnamed: 0	int64	0	[0, 1, 2, 3, 4]
1	customer_id	string	0	[G1606, F8889, C5068, G9820, H7261]
2	gender	string	0	[Female, Female, Female, Male, Male]
3	location	string	0	[Lancashire, Essex, Essex, West Yorkshire, Gre...
4	partner	int32	0	[0, 0, 0, 1, 0]
5	dependents	float64	2208	[1.0, 1.0, nan, 1.0, 1.0]
6	senior	int64	0	[0, 0, 1, 1, 0]
7	Tenure	float64	0	[20.0, 4.0, 9.0, 9.0, 6.0]
8	monthly_cost	string	7271	[<NA>, <NA>, <NA>, <NA>, <NA>]
9	package	int64	0	[2, 1, 2, 4, 2]
10	survey	float64	597	[0.0, 5.0, 0.0, 9.0, 5.0]
11	churn	string	59	[No, No, No, No, No]

Datatypes changed :

Datatypes changed	**Not Changed**
Customer_id	Senior
Gender	tenure
Location	package
Partner	
Dependents	
Monthly_cost	
Survey	
class	

Data Preparation stage ends here we saved our dataset at this stage as **“EDA_ready_data.csv”.**

Now we will perform Exploratory data Analysis on this dataset to extract some important information about the dataset.

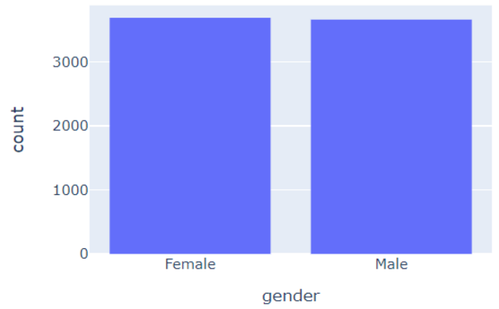
5. Exploratory data Analysis :

5.1 Cleaning the Dataset :

We have plotted histogram for the count of every feature. We have also excluded customer_id column because it is unique for every customer.

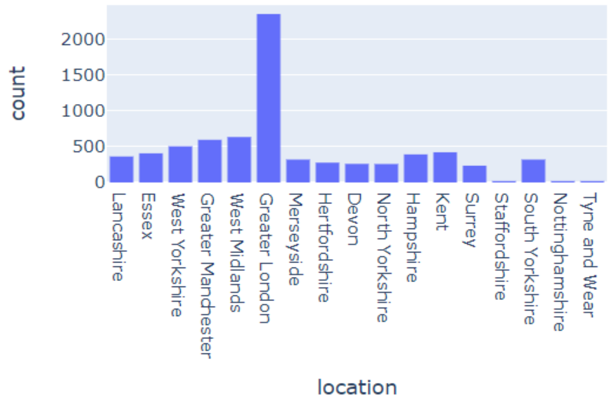
5.2 Insights of Data

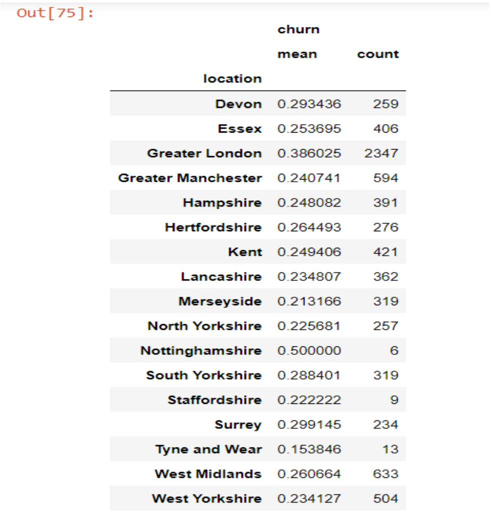
Gender :



We can clearly see that there are more female customers than male customers. Count of female customer is 3689 while count of male customer is 3661. This is important information because in future, Lu’s communication can give offers according to this information.

Location:

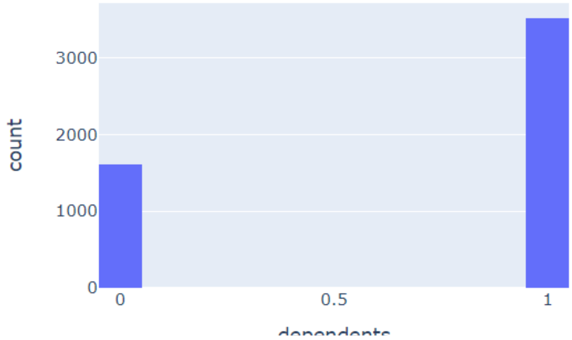




Out[75]:

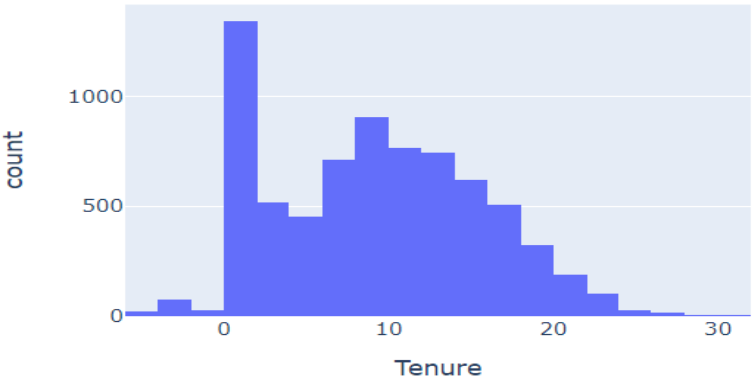
	churn	
	mean	count
location		
Devon	0.293436	259
Essex	0.253695	406
Greater London	0.386025	2347
Greater Manchester	0.240741	594
Hampshire	0.248082	391
Hertfordshire	0.264493	276
Kent	0.249406	421
Lancashire	0.234807	362
Merseyside	0.213166	319
North Yorkshire	0.225681	257
Nottinghamshire	0.500000	6
South Yorkshire	0.288401	319
Staffordshire	0.222222	9
Surrey	0.299145	234
Tyne and Wear	0.153846	13
West Midlands	0.260664	633
West Yorkshire	0.234127	504

As we can clearly see, Greater London has most customers and highest churn rate as compared to any other region in UK. Similarly, Manchester, West Midlands, West Yorkshire are having similar churn rate. These regions Lu’s Communication should focus on.



Dependents :
Around 3531 customers have their dependents with them and 1611 customers don't have dependents living with them.

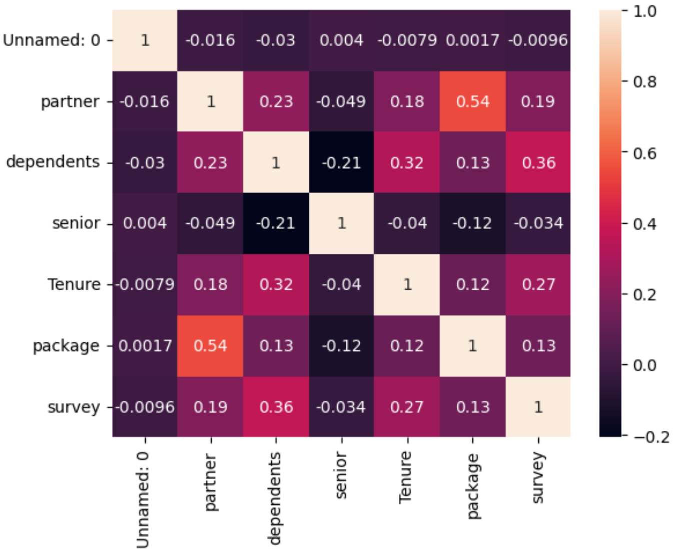
Tenure :



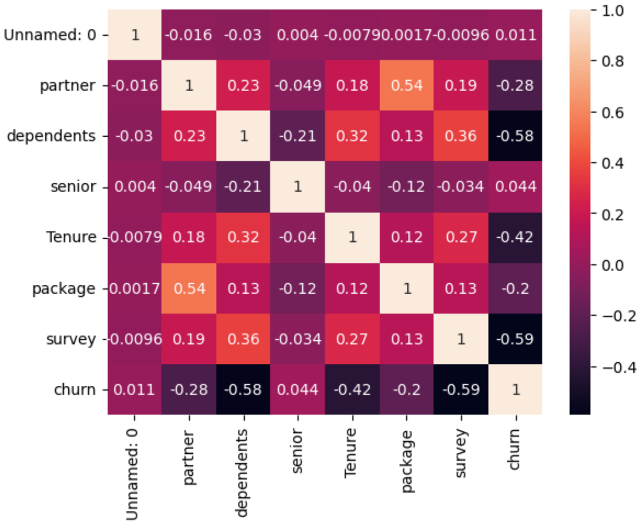
We can clearly see that, there are 1347 customers between tenure 0-2 years followed by 904 customers with tenure between 8-10 years. Notably, there are 2 outliers in tenure with 29 years and 30 years respectively. Both of these numbers can present the possibility so we decided that we will not remove these outliers and it could be an actual data.

Correlation :

Without Churn Column



with Churn column



6. Initial Hypotheses

Without Churn Feature :

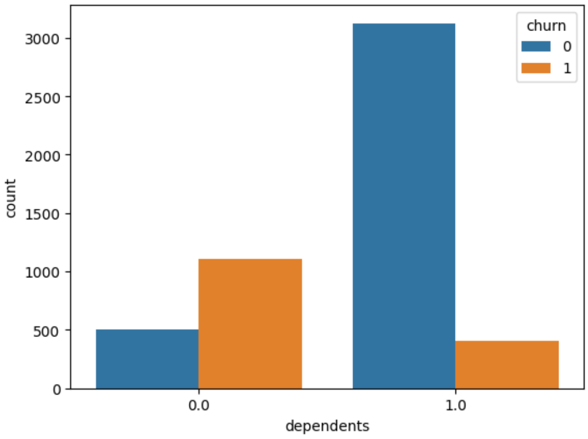
1. It is clearly seen that partner feature is having positive corelation with package feature. So we compared both the features and we got important insight that If there is no partner then customers are tend to buy the cheap package but If there are partners then customers are inclined towards costliest package.

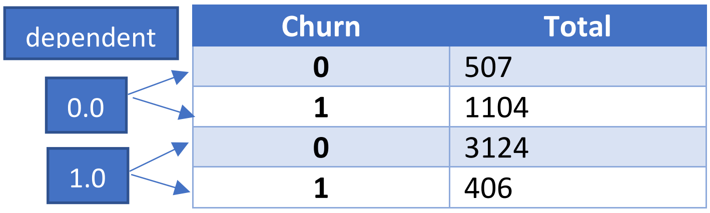
With Churn Feature :

From the corelation on the right, its clearly seen that the churn column has high corelation with dependents, Tenure and Surevy.

2. Do Dependent affect on churn rate?

We got some useful information from this analysis. Here we have changed values of churn where **No = 0 and Yes = 1.**

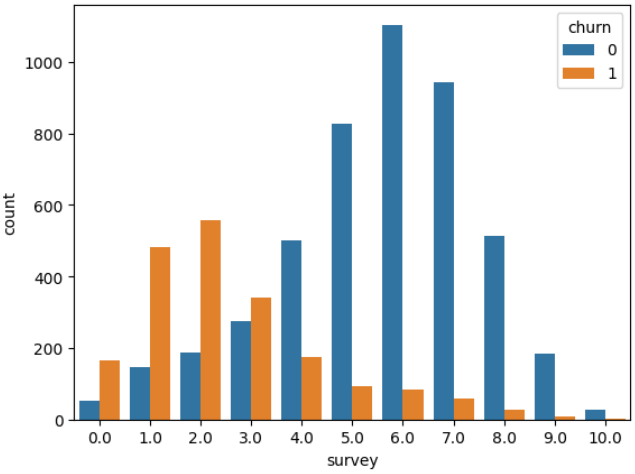




dependent	Churn	Total
0.0	**0**	507
	1	1104
1.0	**0**	3124
	1	406

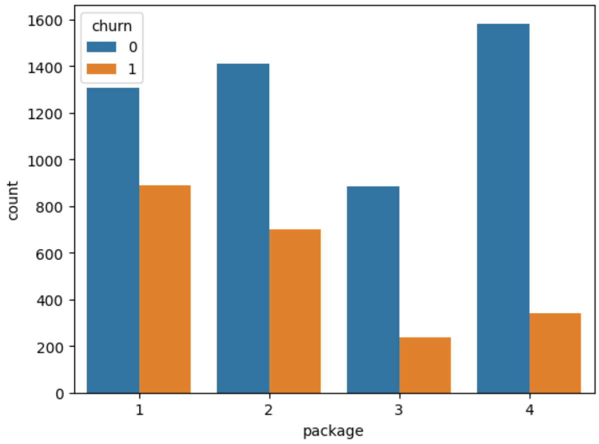
If there are dependents to the customer then they are less likely to churn than that of non-dependents.

3. Is Low rating by user survey are responsible for increase in churn rate:



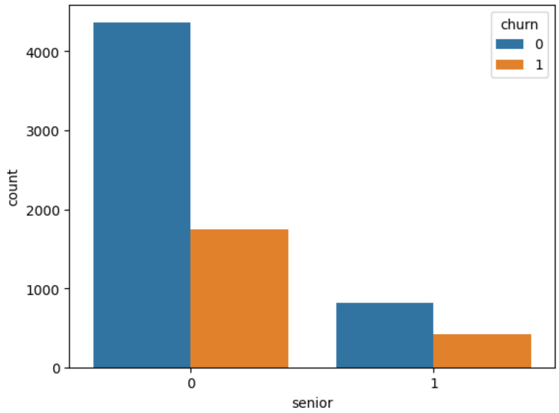
We have seen that less survey score results into churn rate. Less score referring to high churn rate. As we can see, survey score 2 out of 10 is referring to high churn rate. 556 customers who have given survey score 2 out of 10 has been churned. Lu's communication should be focusing more on the customers who have given survey score between 0 to 4.Customers who have given score between 8 to 10 are most satisfied with the services provided by Lu's Communication hence less churn rate is observed in the plot.

4. Did Customer who subscribe for lower package churns on high rate?



It has been seen that customers who have chosen less package are churned a lot. Package 1 and Package 2 are having high churn rate. Lu's communication needs to focus more on the customers who have signed up for Package 1 and Package 2. Similarly, Package 3 is having less churn rate. Finally Package 4 is also having higher churn rate but it is less as compared to customer count vs churn rate ratio.

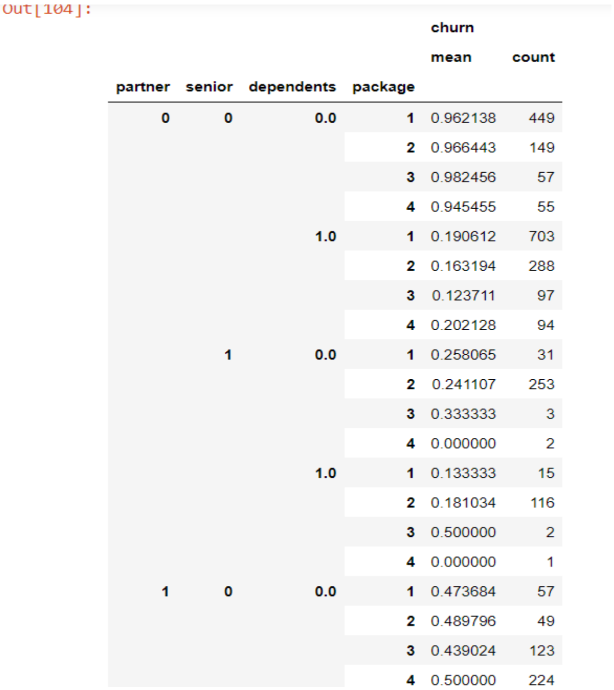
5. If Customer is senior will he increase the churn rate?



In the plot, blue bar represents No churn while yellow bar shows churned customer. We can clearly see that, if customer is associated with any elderly, then they are most likely to churn. This information could play a role in Lu's Communication strategy plan to avoid churn of customers.

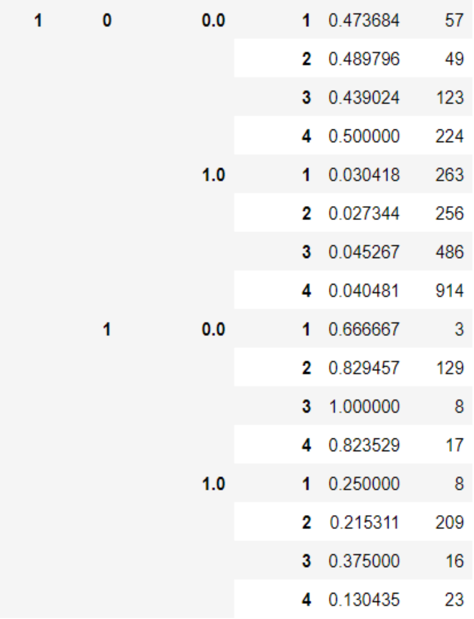
6. Can we get any important information by grouping Partner, senior , dependents and package feature.

We combined partner, senior, dependents and package feature so that we can get important insights with churn rate, following is a result that we got from it.



Out[104]:

				churn	
				mean	count
partner	senior	dependents	package		
0	0	0.0	1	0.962138	449
			2	0.966443	149
			3	0.982456	57
			4	0.945455	55
		1.0	1	0.190612	703
			2	0.163194	288
			3	0.123711	97
			4	0.202128	94
	1	0.0	1	0.258065	31
			2	0.241107	253
			3	0.333333	3
			4	0.000000	2
		1.0	1	0.133333	15
			2	0.181034	116
			3	0.500000	2
			4	0.000000	1
1	0	0.0	1	0.473684	57
			2	0.489796	49
			3	0.439024	123
			4	0.500000	224



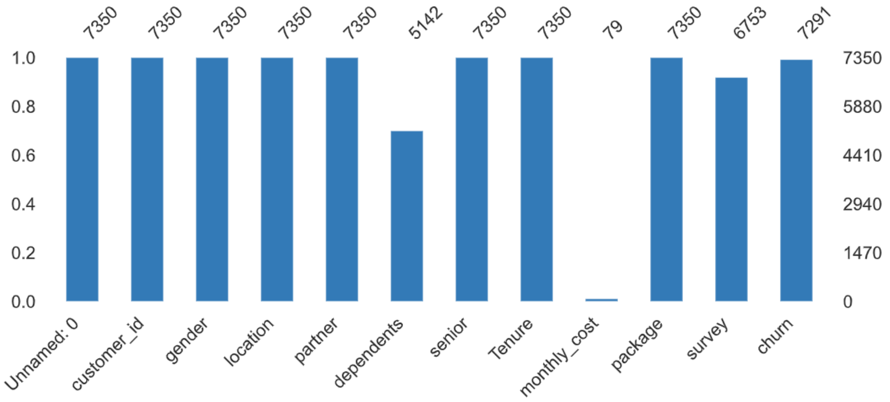
1	0	0.0	1	0.473684	57
			2	0.489796	49
			3	0.439024	123
			4	0.500000	224
		1.0	1	0.030418	263
			2	0.027344	256
			3	0.045267	486
			4	0.040481	914
	1	0.0	1	0.666667	3
			2	0.829457	129
			3	1.000000	8
			4	0.823529	17
		1.0	1	0.250000	8
			2	0.215311	209
			3	0.375000	16
			4	0.130435	23

1. Those customers who don’t have any partner, senior or dependent associated with them leads to churn. Almost 449 customers churned in this way which rounds up to almost 96% from total population of this combination. In simple terms, Lu’s Communication should focus more on those customers who are not associated with partner, senior or any dependents.

2. those customer don’t have partner but have senior and dependents have shown less churn rate.

7. Data Processing :

7.1 Dealing with Missing values :



Feature Name	Missing Values
dependents	2208
monthly_cost	7271
Survey	597
churn	59

In the right table above, We have specified the missing values from given features. We don’t need Customer_id Column therefore we have dropped it from the dataset.

1. churn :

There are 59 missing values in churn so we decided to remove those records.

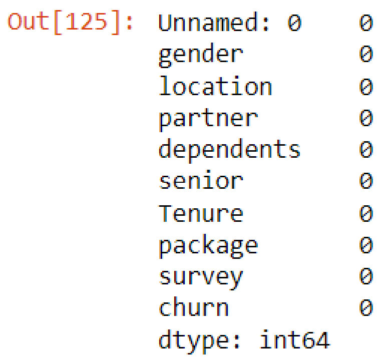
2. dependents & Survey :

Now we can see we have 2191 null records in dependent column and 597 records in Survey column, we can not ignore them since they are correlated to churn hence we are going to fill them by using KNN Imputation. The idea in KNN methods is to identify 'k' samples in the dataset that are similar or close in the space. Then we use these 'k' samples to estimate the value of the missing data points. Each sample's missing values are imputed using the mean value of the 'k'-neighbors found in the dataset.

3. monthly cost :

Monthly_cost have maximum number of missing values. We will calculate a monthly cost values which is associated wit each package. Then we will multiply that value with 2% discount on each year of tenure and we will subtract that value from actual monthly_cost and result will be stored in monthly_cost but tenure cannot be more than 25 years.
For example :
If Package 1 value is 26 pounds and tenure is 2 years then,
Monthly_cost = 26 – (26 * 0.02 * 24)
Here, 24 refers to months. 2 * 12 = 24

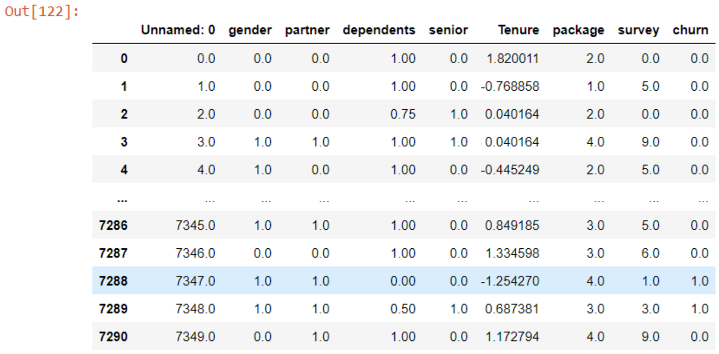
Here is a result after dealing with missing values.



```
Out[125]: Unnamed: 0    0
          gender        0
          location      0
          partner       0
          dependents    0
          senior        0
          Tenure        0
          package       0
          survey        0
          churn         0
          dtype: int64
```

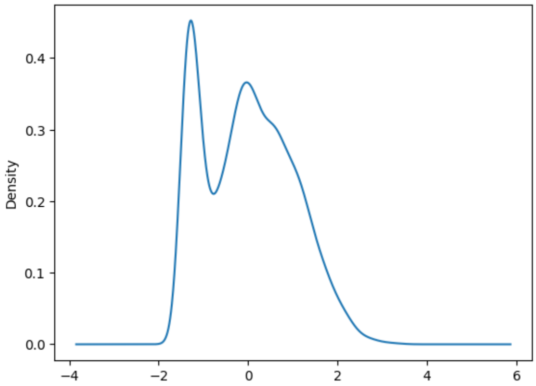
7.2 Dealing with outliers :

An outlier is an observation that lies an abnormal distance from other values in a random sample from a population. In EDA we found only Tenure column have outliers which are in very less number as compared to whole dataset and it contains negative value and tenure canot be negative so here we are taking abs values for tenure where it is negative.

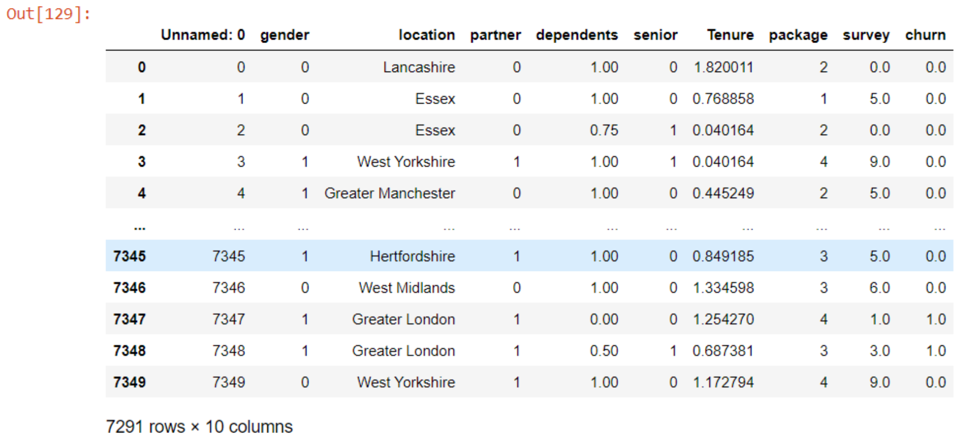


Out[122]:

	Unnamed: 0	gender	partner	dependents	senior	Tenure	package	survey	churn
0	0.0	0.0	0.0	1.00	0.0	1.820011	2.0	0.0	0.0
1	1.0	0.0	0.0	1.00	0.0	-0.768858	1.0	5.0	0.0
2	2.0	0.0	0.0	0.75	1.0	0.040164	2.0	0.0	0.0
3	3.0	1.0	1.0	1.00	1.0	0.040164	4.0	9.0	0.0
4	4.0	1.0	0.0	1.00	0.0	-0.445249	2.0	5.0	0.0
...	...	...	...	...	...	...	...	...	...
7286	7345.0	1.0	1.0	1.00	0.0	0.849185	3.0	5.0	0.0
7287	7346.0	0.0	0.0	1.00	0.0	1.334598	3.0	6.0	0.0
7288	7347.0	1.0	1.0	0.00	0.0	-1.254270	4.0	1.0	1.0
7289	7348.0	1.0	1.0	0.50	1.0	0.687381	3.0	3.0	1.0
7290	7349.0	0.0	1.0	1.00	0.0	1.172794	4.0	9.0	0.0



After taking absolute value



Out[129]:

	Unnamed: 0	gender	location	partner	dependents	senior	Tenure	package	survey	churn
0	0	0	Lancashire	0	1.00	0	1.820011	2	0.0	0.0
1	1	0	Essex	0	1.00	0	0.768858	1	5.0	0.0
2	2	0	Essex	0	0.75	1	0.040164	2	0.0	0.0
3	3	1	West Yorkshire	1	1.00	1	0.040164	4	9.0	0.0
4	4	1	Greater Manchester	0	1.00	0	0.445249	2	5.0	0.0
...	...	...	...	...	...	...	...	...	...	...
7345	7345	1	Hertfordshire	1	1.00	0	0.849185	3	5.0	0.0
7346	7346	0	West Midlands	0	1.00	0	1.334598	3	6.0	0.0
7347	7347	1	Greater London	1	0.00	0	1.254270	4	1.0	1.0
7348	7348	1	Greater London	1	0.50	1	0.687381	3	3.0	1.0
7349	7349	0	West Yorkshire	1	1.00	0	1.172794	4	9.0	0.0

7291 rows × 10 columns

7.3 Dealing with different scales of data (feature encoding) :

As there are many categories inside the **location** column there is not a best way to encode it. We will use two approaches :

1. One Hot encoding :

One hot encoding is a method of encoding in which categorical variable is transformed into a form which will be further sent to machine learning algorithm for better prediction. We performed One hot encoding and we stored the data in onehot_df dataset.

2. Weight of evidence encoding

Weight of Evidence (WoE) measures the **"**strength" of a grouping technique to separate good and bad. This method was developed primarily to build a predictive model to evaluate the risk of loan default in the credit and financial industry. Weight of evidence (WOE) measures how much the evidence supports or undermines a hypothesis**.** It is computed as below:

$$WoE = \left[ln \left(\frac{Distr\ Goods}{Distr\ Bads} \right) \right] * 100$$

We performed Weight of encoding and stored results in wef_df dataset. (Roy, 2021)

8. Feature Selection :

8.1 Principle component analysis :

Linear dimensionality reduction using Singular Value Decomposition of the data to project it to a lower dimensional space. The input data is centered but not scaled for each feature before applying the SVD. Its idea is simple—reduce the dimensionality of a dataset, while preserving as much 'variability' (i.e. statistical information) as possible. . (Jolliffe and Cadima, 2016)

Although our dataset is relatively small, let's use PCA for feature selection and see if it improves our accuracy. We first dropped our dependent column which is churn. For selection of components we decided to calculate explained_variance_ratio.

Explained variance is a statistical measure of how much variation in a dataset can be attributed to each of the principal components (eigenvectors) generated by a PCA**.** In very basic terms, it refers to the amount of variability in a data set that can be attributed to each individual principal component. (Kumar, 2020)

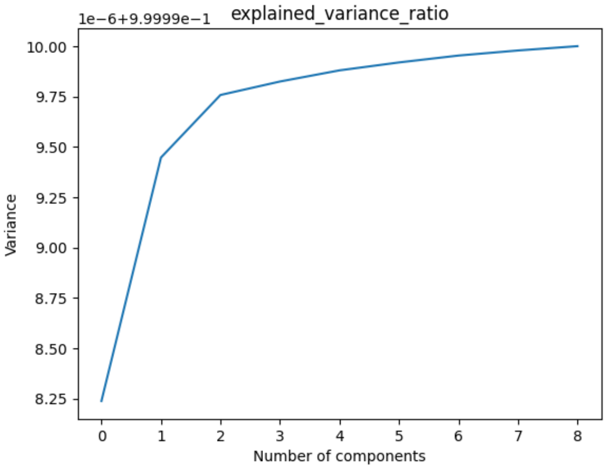


Figure : PCA of Weight of Evidence

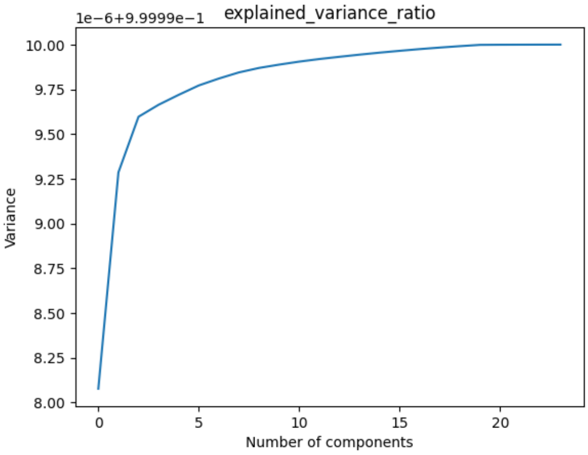


Figure : PCA of One-hot-Encoding

For weight of evidence encoding, We can see that around 95% of variance is being explained by 5 components. So instead of giving all 8 columns as input in our algorithm let's use these principle components instead.

In second analysis, We can see that around 95% of variance is being explained by 10 components. So instead of giving all 20 columns as input in our algorithm let's use these principle components instead.

We stored these 10 components in principal_onehot_df dataset.

9. Data Modeling :

We decided to use following machine learning models :

- Decision tree
- XGBoost
- Random Forest
- SVM
- Logistic Rgression

9.1 Machine Learning Models :

Accuracy Score :
Accuracy classification score method computes subset accuracy in multilevel classification.
Precision Score :
Precision quantifies the number of positive class predictions that actually belong to the positive class.
Recall Score :
Recall quantifies the number of positive class predictions made out of all positive examples in the dataset.
F-1 Score :
F-Measure provides a single score that balances both the concerns of precision and recall in one number. (Brownlee, 2020)

9.1.1 Decision Tree :

Decision Trees (DTs) are a non-parametric supervised learning method used for classification and regression. The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features. A tree can be seen as a piecewise constant approximation. (scikit learn, 2009)

One Hot Encoding

Parameters	Values
Accuracy score	0.83
Precision score	0.69
Recall score	0.72
F-1 Score	0.71

Weight of Evidence Encoding

Parameters	Values
Accuracy score	0.82
Precision score	0.68
Recall score	0.69
F-1 Score	0.69

9.1.2 Xgboost :

XGBoost is an optimized distributed gradient boosting library designed to be highly efficient, flexible and portable. It implements machine learning algorithms under the Gradient Boosting framework. XGBoost provides a parallel tree boosting (also known as GBDT, GBM) that solve many data science problems in a fast and accurate way. The same code runs on major distributed environment (Hadoop, SGE, MPI) and can solve problems beyond billions of examples. (XGBoost, 2021)

One Hot Encoding

Parameters	Values
Accuracy score	0.87
Precision score	0.81
Recall score	0.75
F-1 Score	0.72

Weight Of Evidence Encoding

Parameters	Values
Accuracy score	0.86
Precision score	0.78
Recall score	0.72
F-1 Score	0.75

9.1.3 random forest:

Random forest is a decision tree algorithm which combines output of multiple decision trees to get results. This algorithm is really flexible and easy to use. (IBM, 2020)

One Hot Encoding

Parameters	Values
Accuracy score	0.88
Precision score	0.82
Recall score	0.75
F-1 Score	0.78

Weight of Evidence Encoding

Parameters	Values
Accuracy score	0.86
Precision score	0.79
Recall score	0.73
F-1 Score	0.76

9.1.4 SVM (Support Vector machine) :

Support vector machines (SVMs) are really powerful and flexible class of supervised algorithms. They are used for both classification and regression.

One Hot Encoding

Parameters	Values
Accuracy score	0.711
Precision score	0
Recall score	0
F-1 Score	0

Weight Of Evidence Encoding

Parameters	Values
Accuracy score	0.711
Precision score	0
Recall score	0
F-1 Score	0

9.1.5 Logistic Regression :

Logistic regression models the probabilities for classification problems with two possible outcomes. It's an extension of the linear regression model for classification problems.

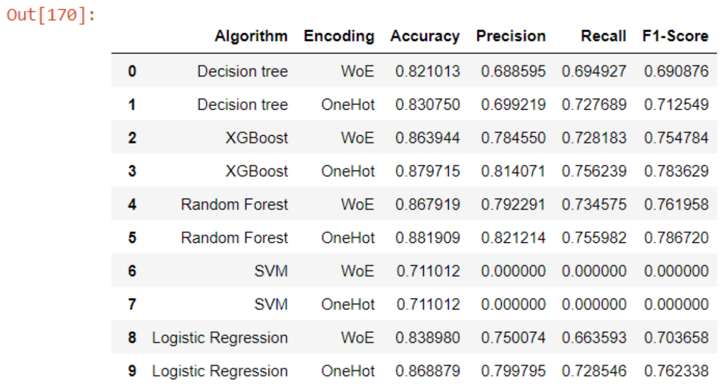
One Hot Encoding

Parameters	Values
Accuracy score	0.86
Precision score	0.79
Recall score	0.72
F-1 Score	0.76

Weight Of Evidence Encoding

Parameters	Values
Accuracy score	0.83
Precision score	0.75
Recall score	0.72
F-1 Score	0.70

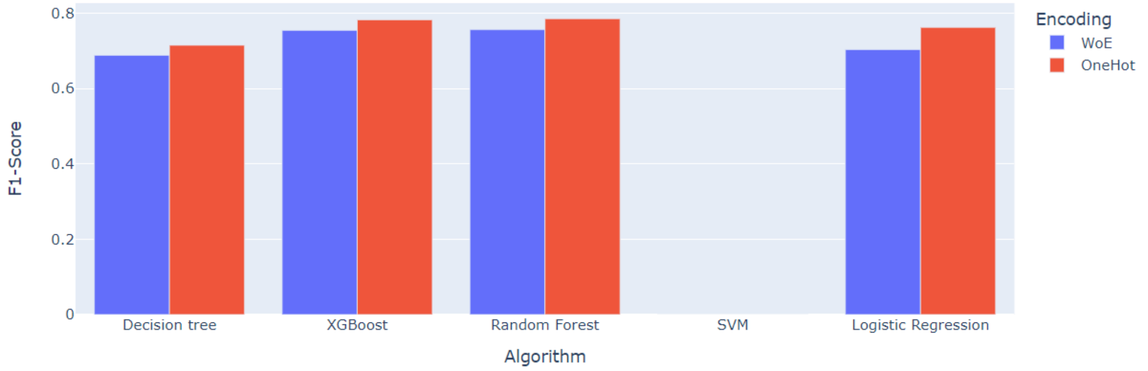
10. Conclusion :

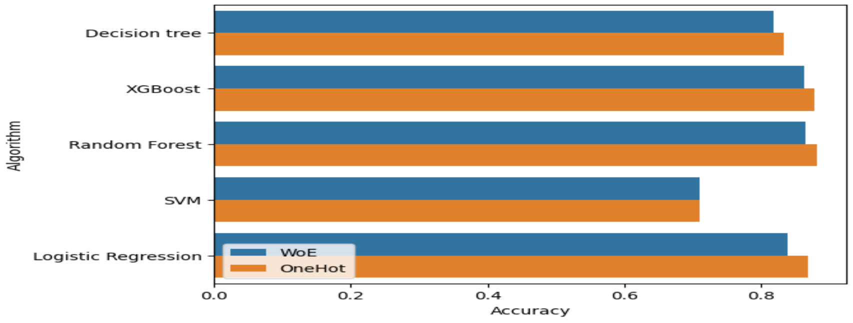


Out[170]:

	Algorithm	Encoding	Accuracy	Precision	Recall	F1-Score
0	Decision tree	WoE	0.821013	0.688595	0.694927	0.690876
1	Decision tree	OneHot	0.830750	0.699219	0.727689	0.712549
2	XGBoost	WoE	0.863944	0.784550	0.728183	0.754784
3	XGBoost	OneHot	0.879715	0.814071	0.756239	0.783629
4	Random Forest	WoE	0.867919	0.792291	0.734575	0.761958
5	Random Forest	OneHot	0.881909	0.821214	0.755982	0.786720
6	SVM	WoE	0.711012	0.000000	0.000000	0.000000
7	SVM	OneHot	0.711012	0.000000	0.000000	0.000000
8	Logistic Regression	WoE	0.838980	0.750074	0.663593	0.703658
9	Logistic Regression	OneHot	0.868879	0.799795	0.728546	0.762338

We evaluated 5 models across favoured performance metrics like Accuracy, Precision, Recall, and F1 Score. The plot below illustrates the performance of all the considered models. From the findings, we can conclude that the Random Forest algorithm gave the best performance using One Hot encoding.





Random Forest algorithm gives 88% accuracy which is highest as compared to Decision tree, XGBoost, SVM and Logistic Regression. Second best is XGBoost One hot encoding at 87% accuracy.

11.References :

1. Roy, B. (2021). *All about Categorical Variable Encoding*. [online] Medium. Available at: https://towardsdatascience.com/all-about-categorical-variable-encoding-305f3361fd02#:~:text=implement%20as%20below%3A- [Accessed 7 Aug. 2022].

2. SearchBusinessAnalytics. (n.d.). *What is Data Preparation? An In-Depth Guide to Data Prep*. [online] Available at: https://www.techtarget.com/searchbusinessanalytics/definition/data-preparation.

3. Jolliffe, I.T. and Cadima, J. (2016). Principal component analysis: a review and recent developments. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 374(2065), p.20150202. doi:10.1098/rsta.2015.0202.

4. Kumar, A. (2020). *PCA Explained Variance Concepts with Python Example*. [online] Data Analytics. Available at: https://vitalflux.com/pca-explained-variance-concept-python-example/.

5. scikit learn (2009). *1.10. Decision Trees — scikit-learn 0.22 documentation*. [online] Scikit-learn.org. Available at: https://scikit-learn.org/stable/modules/tree.html.

6. XGBoost (2021). *XGBoost Documentation — xgboost 1.5.1 documentation*. [online] xgboost.readthedocs.io. Available at: https://xgboost.readthedocs.io/en/stable/.

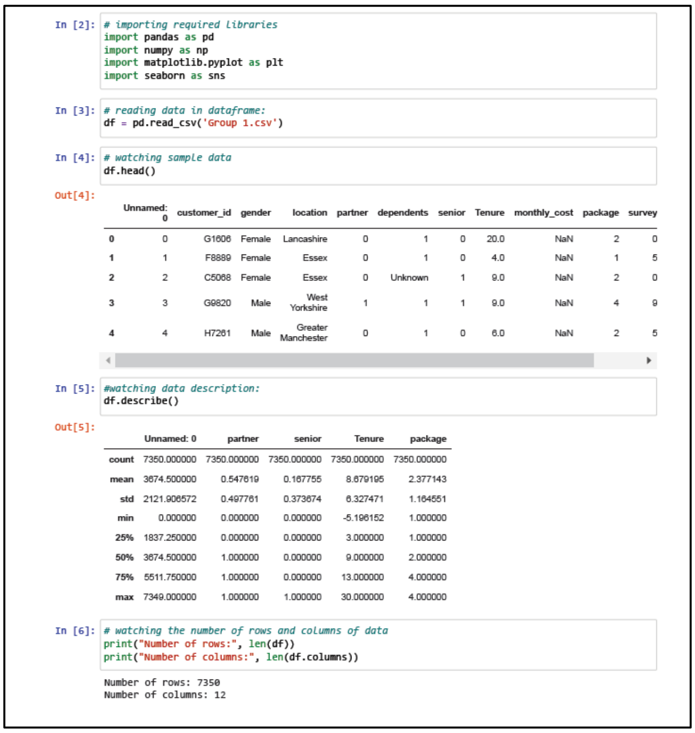
7. IBM (2020). *What is Random Forest?* [online] www.ibm.com. Available at: https://www.ibm.com/cloud/learn/random-forest#:~:text=Random%20forest%20is%20a%20commonly [Accessed 7 Aug. 2022].

8. Python data Science handbook (n.d.). *In-Depth: Support Vector Machines | Python Data Science Handbook*. [online] jakevdp.github.io. Available at: https://jakevdp.github.io/PythonDataScienceHandbook/05.07-support-vector-machines.html.

9. Brownlee, J. (2020). *How to Calculate Precision, Recall, and F-Measure for Imbalanced Classification*. [online] Machine Learning Mastery. Available at: https://machinelearningmastery.com/precision-recall-and-f-measure-for-imbalanced-classification/#:~:text=Precision%20quantifies%20the%20number%20of.

Appendix :

Few Screenshots of code.



```
In [2]: # importing required libraries
        import pandas as pd
        import numpy as np
        import matplotlib.pyplot as plt
        import seaborn as sns

In [3]: # reading data in dataframe:
        df = pd.read_csv('Group 1.csv')

In [4]: # watching sample data
        df.head()
```

Out[4]:

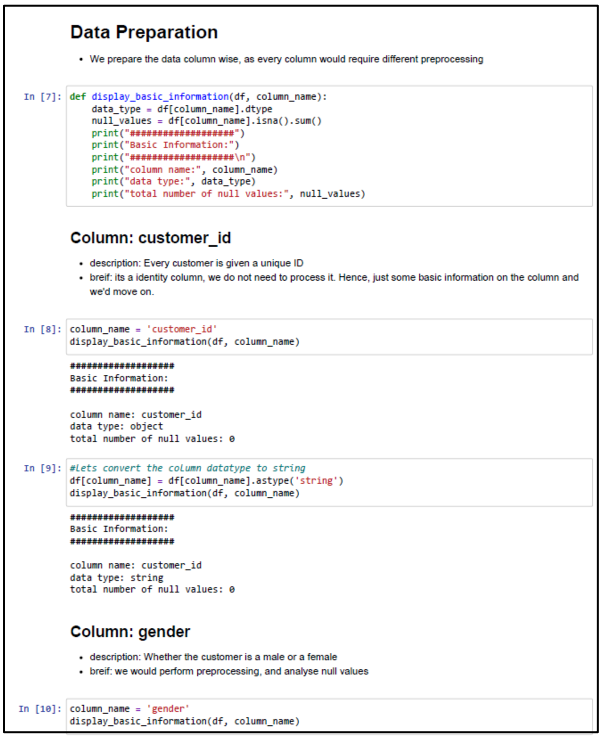
	Unnamed: 0	customer_id	gender	location	partner	dependents	senior	Tenure	monthly_cost	package	survey
0	0	G1606	Female	Lancashire	0	1	0	20.0	NaN	2	0
1	1	F8889	Female	Essex	0	1	0	4.0	NaN	1	5
2	2	C5068	Female	Essex	0	Unknown	1	9.0	NaN	2	0
3	3	G9820	Male	West Yorkshire	1	1	1	9.0	NaN	4	9
4	4	H7261	Male	Greater Manchester	0	1	0	6.0	NaN	2	5

```
In [5]: #watching data description:
        df.describe()
```

Out[5]:

	Unnamed: 0	partner	senior	Tenure	package
count	7350.000000	7350.000000	7350.000000	7350.000000	7350.000000
mean	3674.500000	0.547619	0.167755	8.679195	2.377143
std	2121.906572	0.497761	0.373674	6.327471	1.164551
min	0.000000	0.000000	0.000000	-5.196152	1.000000
25%	1837.250000	0.000000	0.000000	3.000000	1.000000
50%	3674.500000	1.000000	0.000000	9.000000	2.000000
75%	5511.750000	1.000000	0.000000	13.000000	4.000000
max	7349.000000	1.000000	1.000000	30.000000	4.000000

```
In [6]: # watching the number of rows and columns of data
        print("Number of rows:", len(df))
        print("Number of columns:", len(df.columns))

        Number of rows: 7350
        Number of columns: 12
```



Data Preparation

- We prepare the data column wise, as every column would require different preprocessing

```
In [7]: def display_basic_information(df, column_name):
            data_type = df[column_name].dtype
            null_values = df[column_name].isna().sum()
            print("###################")
            print("Basic Information:")
            print("###################\n")
            print("column name:", column_name)
            print("data type:", data_type)
            print("total number of null values:", null_values)
```

Column: customer_id

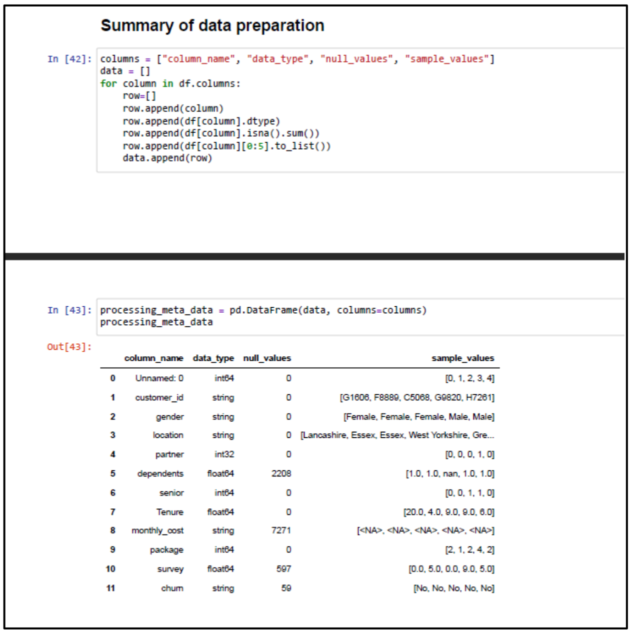
- description: Every customer is given a unique ID
- breif: its a identity column, we do not need to process it. Hence, just some basic information on the column and we'd move on.

```
In [8]: column_name = 'customer_id'
        display_basic_information(df, column_name)

        ###################
        Basic Information:
        ###################

        column name: customer_id
        data type: object
        total number of null values: 0

In [9]: #Lets convert the column datatype to string
        df[column_name] = df[column_name].astype('string')
        display_basic_information(df, column_name)

        ###################
        Basic Information:
        ###################

        column name: customer_id
        data type: string
        total number of null values: 0
```

Column: gender

- description: Whether the customer is a male or a female
- breif: we would perform preprocessing, and analyse null values

```
In [10]: column_name = 'gender'
         display_basic_information(df, column_name)
```

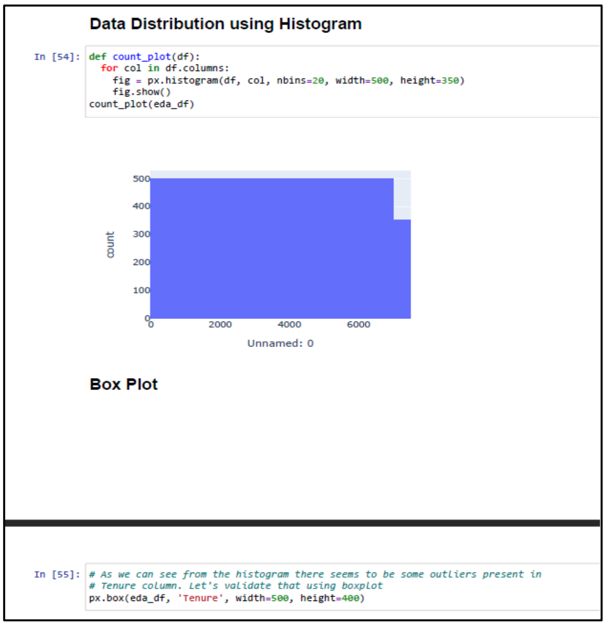


Summary of data preparation

```
In [42]: columns = ["column_name", "data_type", "null_values", "sample_values"]
         data = []
         for column in df.columns:
             row=[]
             row.append(column)
             row.append(df[column].dtype)
             row.append(df[column].isna().sum())
             row.append(df[column][0:5].to_list())
             data.append(row)

In [43]: processing_meta_data = pd.DataFrame(data, columns=columns)
         processing_meta_data
```

Out[43]:

	column_name	data_type	null_values	sample_values
0	Unnamed: 0	int64	0	[0, 1, 2, 3, 4]
1	customer_id	string	0	[G1606, F8889, C5068, G9820, H7261]
2	gender	string	0	[Female, Female, Female, Male, Male]
3	location	string	0	[Lancashire, Essex, Essex, West Yorkshire, Gre...
4	partner	int32	0	[0, 0, 0, 1, 0]
5	dependents	float64	2208	[1.0, 1.0, nan, 1.0, 1.0]
6	senior	int64	0	[0, 0, 1, 1, 0]
7	Tenure	float64	0	[20.0, 4.0, 9.0, 9.0, 6.0]
8	monthly_cost	string	7271	[<NA>, <NA>, <NA>, <NA>, <NA>]
9	package	int64	0	[2, 1, 2, 4, 2]
10	survey	float64	597	[0.0, 5.0, 0.0, 9.0, 5.0]
11	churn	string	59	[No, No, No, No, No]



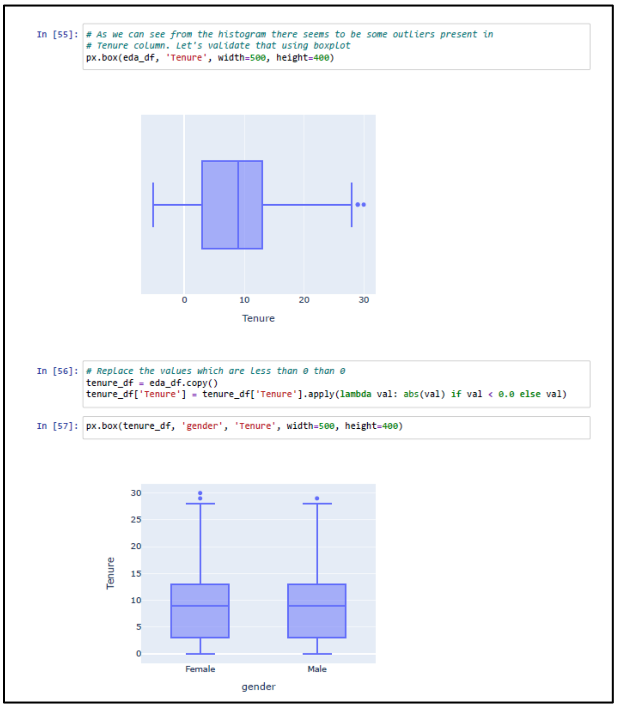
Data Distribution using Histogram

```
In [54]: def count_plot(df):
           for col in df.columns:
             fig = px.histogram(df, col, nbins=20, width=500, height=350)
             fig.show()
         count_plot(eda_df)
```

Box Plot

```
In [55]: # As we can see from the histogram there seems to be some outliers present in
         # Tenure column. Let's validate that using boxplot
         px.box(eda_df, 'Tenure', width=500, height=400)
```

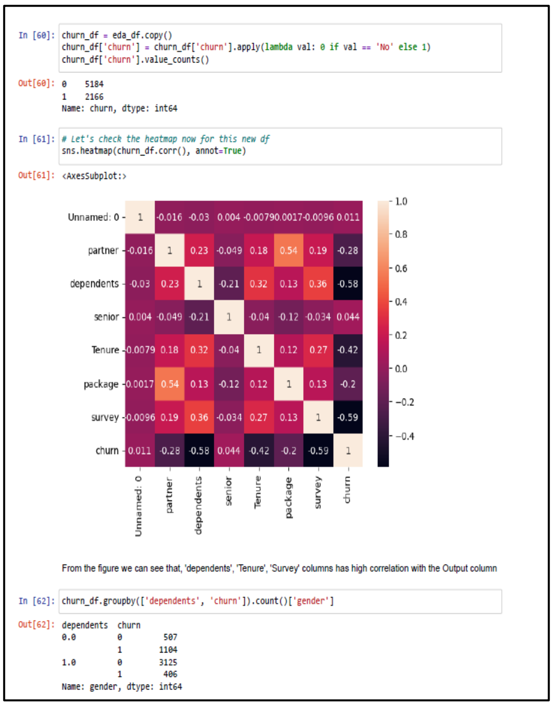
Exploratory data Analysis :



```
In [55]: # As we can see from the histogram there seems to be some outliers present in
         # Tenure column. Let's validate that using boxplot
         px.box(eda_df, 'Tenure', width=500, height=400)
```

```
In [56]: # Replace the values which are less than 0 than 0
         tenure_df = eda_df.copy()
         tenure_df['Tenure'] = tenure_df['Tenure'].apply(lambda val: abs(val) if val < 0.0 else val)

In [57]: px.box(tenure_df, 'gender', 'Tenure', width=500, height=400)
```

30
25
20
15
10
5
0
Tenure
Female
Male
gender

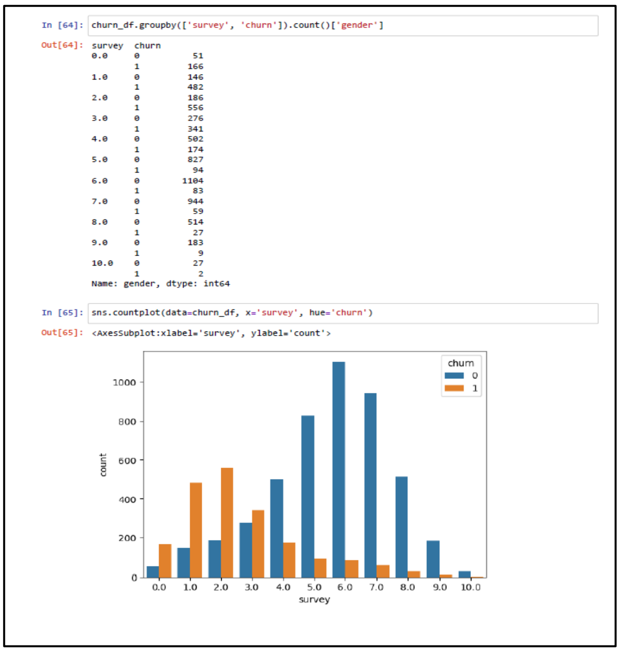


```
In [60]: churn_df = eda_df.copy()
         churn_df['churn'] = churn_df['churn'].apply(lambda val: 0 if val == 'No' else 1)
         churn_df['churn'].value_counts()

Out[60]: 0    5184
         1    2166
         Name: churn, dtype: int64

In [61]: # Let's check the heatmap now for this new df
         sns.heatmap(churn_df.corr(), annot=True)

Out[61]: <AxesSubplot:>
```

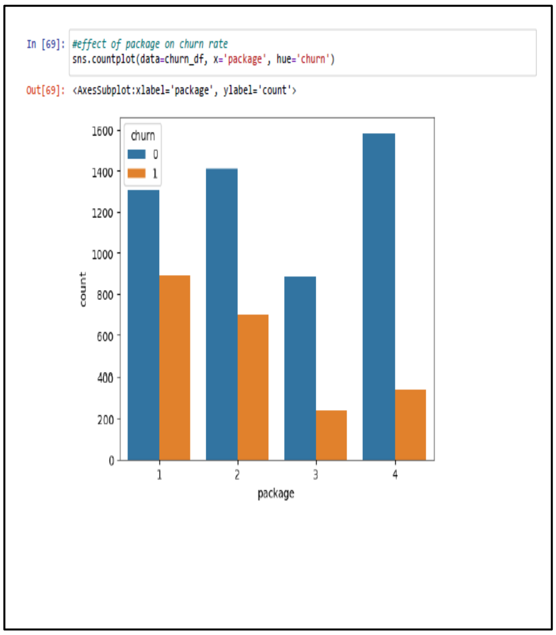
Unnamed: 0 1 -0.016 -0.03 0.004 -0.0079 0.0017 -0.0096 0.011
partner -0.016 1 0.23 -0.049 0.18 0.54 0.19 -0.28
dependents -0.03 0.23 1 -0.21 0.32 0.13 0.36 -0.58
senior 0.004 -0.049 -0.21 1 -0.04 -0.12 -0.034 0.044
Tenure -0.0079 0.18 0.32 -0.04 1 0.12 0.27 -0.42
package 0.0017 0.54 0.13 -0.12 0.12 1 0.13 -0.2
survey -0.0096 0.19 0.36 -0.034 0.27 0.13 1 -0.59
churn 0.011 -0.28 -0.58 0.044 -0.42 -0.2 -0.59 1
Unnamed: 0
partner
dependents
senior
Tenure
package
survey
churn
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4

From the figure we can see that, 'dependents', 'Tenure', 'Survey' columns has high correlation with the Output column

```
In [62]: churn_df.groupby(['dependents', 'churn']).count()['gender']

Out[62]: dependents  churn
         0.0         0         507
                     1        1104
         1.0         0        3125
                     1         406
         Name: gender, dtype: int64
```

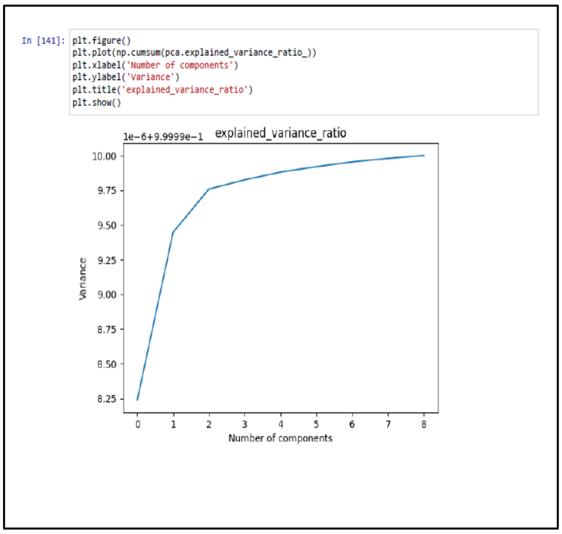


```
In [64]: churn_df.groupby(['survey', 'churn']).count()['gender']

Out[64]: survey  churn
         0.0     0          51
                 1         166
         1.0     0         146
                 1         482
         2.0     0         186
                 1         556
         3.0     0         276
                 1         341
         4.0     0         502
                 1         174
         5.0     0         827
                 1          94
         6.0     0        1104
                 1          83
         7.0     0         944
                 1          59
         8.0     0         514
                 1          27
         9.0     0         183
                 1           9
         10.0    0          27
                 1           2
         Name: gender, dtype: int64

In [65]: sns.countplot(data=churn_df, x='survey', hue='churn')

Out[65]: <AxesSubplot:xlabel='survey', ylabel='count'>
```

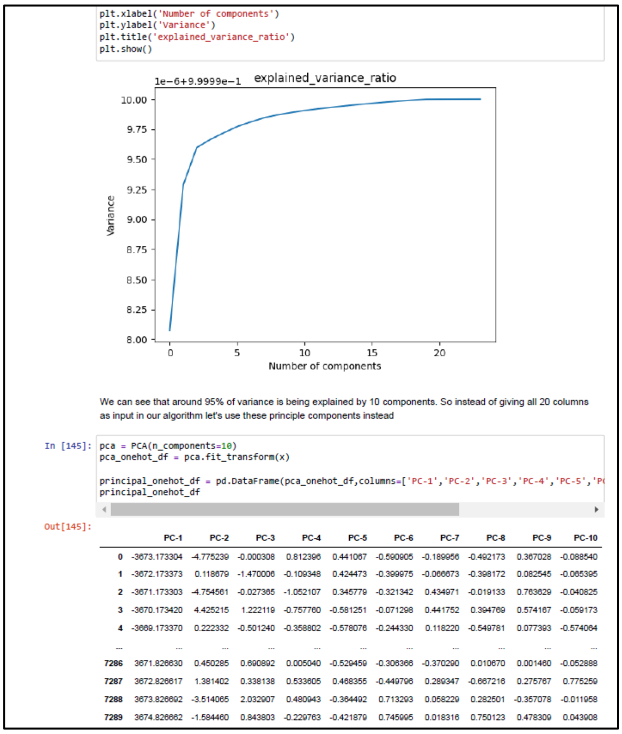


```
In [69]: #effect of package on churn rate
         sns.countplot(data=churn_df, x='package', hue='churn')

Out[69]: <AxesSubplot:xlabel='package', ylabel='count'>
```

Principal Component analysis for one hot Encoding and Woe :



```
In [141]: plt.figure()
          plt.plot(np.cumsum(pca.explained_variance_ratio_))
          plt.xlabel('Number of components')
          plt.ylabel('Variance')
          plt.title('explained_variance_ratio')
          plt.show()
```



```
plt.xlabel('Number of components')
plt.ylabel('Variance')
plt.title('explained_variance_ratio')
plt.show()
```

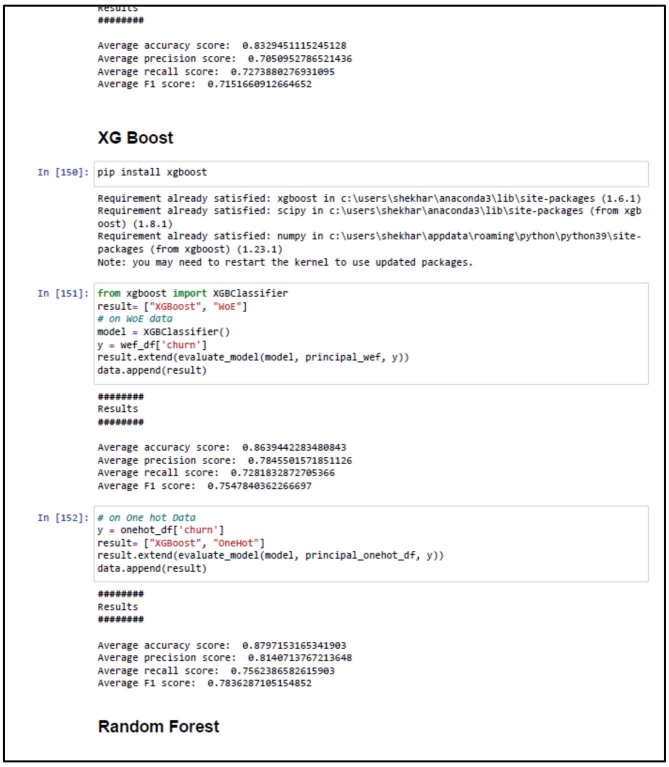
We can see that around 95% of variance is being explained by 10 components. So instead of giving all 20 columns as input in our algorithm let's use these principle components instead

```
In [145]: pca = PCA(n_components=10)
          pca_onehot_df = pca.fit_transform(x)

          principal_onehot_df = pd.DataFrame(pca_onehot_df,columns=['PC-1','PC-2','PC-3','PC-4','PC-5','P(
          principal_onehot_df
```

Out[145]:

	PC-1	PC-2	PC-3	PC-4	PC-5	PC-6	PC-7	PC-8	PC-9	PC-10
0	-3673.173304	-4.775239	-0.000308	0.812396	0.441067	-0.590905	-0.189956	-0.492173	0.367028	-0.088540
1	-3672.173373	0.118679	-1.470006	-0.109348	0.424473	-0.399975	-0.066673	-0.398172	0.082545	-0.065395
2	-3671.173303	-4.754561	-0.027365	-1.052107	0.345779	-0.321342	0.434971	-0.019133	0.763629	-0.040825
3	-3670.173420	4.425215	1.222119	-0.757760	-0.581251	-0.071298	0.441752	0.394769	0.574167	-0.059173
4	-3669.173370	0.222332	-0.501240	-0.358802	-0.578076	-0.244330	0.118220	-0.549781	0.077393	-0.574064
...	...	...	...	...	...	...	...	...	...	...
7286	3671.826630	0.450285	0.690892	0.005040	-0.529459	-0.306366	-0.370290	0.010670	0.001460	-0.052888
7287	3672.826617	1.381402	0.338138	0.533605	0.468355	-0.449796	0.289347	-0.667216	0.275767	0.775259
7288	3673.826692	-3.514065	2.032907	0.480943	-0.364492	0.713293	0.058229	0.282501	-0.357078	-0.011958
7289	3674.826662	-1.584460	0.843803	-0.229763	-0.421879	0.745995	0.018316	0.750123	0.478309	0.043908

Machine Learning Models

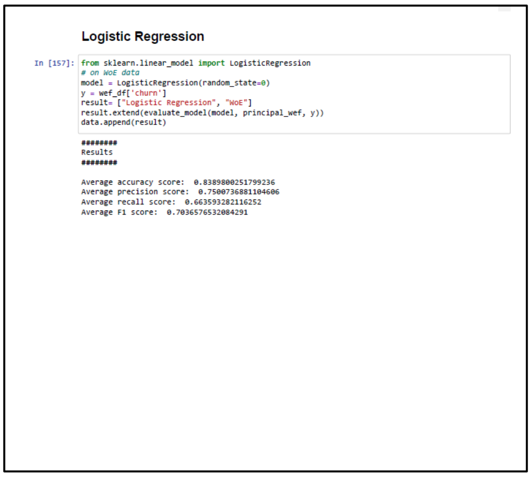


```
Results
########

Average accuracy score:  0.8329451115245128
Average precision score:  0.7050952786521436
Average recall score:  0.7273880276931095
Average F1 score:  0.7151660912664652
```

XG Boost

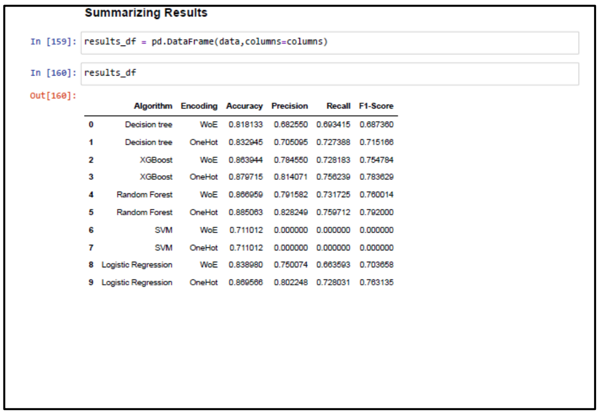
```
In [150]: pip install xgboost

          Requirement already satisfied: xgboost in c:\users\shekhar\anaconda3\lib\site-packages (1.6.1)
          Requirement already satisfied: scipy in c:\users\shekhar\anaconda3\lib\site-packages (from xgb
          oost) (1.8.1)
          Requirement already satisfied: numpy in c:\users\shekhar\appdata\roaming\python\python39\site-
          packages (from xgboost) (1.23.1)
          Note: you may need to restart the kernel to use updated packages.

In [151]: from xgboost import XGBClassifier
          result= ["XGBoost", "WoE"]
          # on WoE data
          model = XGBClassifier()
          y = wef_df['churn']
          result.extend(evaluate_model(model, principal_wef, y))
          data.append(result)

          ########
          Results
          ########

          Average accuracy score:  0.8639442283480843
          Average precision score:  0.7845501571851126
          Average recall score:  0.7281832872705366
          Average F1 score:  0.7547840362266697

In [152]: # on One hot Data
          y = onehot_df['churn']
          result= ["XGBoost", "OneHot"]
          result.extend(evaluate_model(model, principal_onehot_df, y))
          data.append(result)

          ########
          Results
          ########

          Average accuracy score:  0.8797153165341903
          Average precision score:  0.8140713767213648
          Average recall score:  0.7562386582615903
          Average F1 score:  0.7836287105154852
```

Random Forest



Logistic Regression

```
In [157]: from sklearn.linear_model import LogisticRegression
          # on WoE data
          model = LogisticRegression(random_state=0)
          y = wef_df['churn']
          result= ["Logistic Regression", "WoE"]
          result.extend(evaluate_model(model, principal_wef, y))
          data.append(result)

          ########
          Results
          ########

          Average accuracy score:  0.8389800251799236
          Average precision score:  0.7500736881104606
          Average recall score:  0.663593282116252
          Average F1 score:  0.7036576532084291
```



Summarizing Results

```
In [159]: results_df = pd.DataFrame(data,columns=columns)

In [160]: results_df
```

Out[160]:

	Algorithm	Encoding	Accuracy	Precision	Recall	F1-Score
0	Decision tree	WoE	0.818133	0.682550	0.693415	0.687360
1	Decision tree	OneHot	0.832945	0.705095	0.727388	0.715166
2	XGBoost	WoE	0.863944	0.784550	0.728183	0.754784
3	XGBoost	OneHot	0.879715	0.814071	0.756239	0.783629
4	Random Forest	WoE	0.866959	0.791582	0.731725	0.760014
5	Random Forest	OneHot	0.885063	0.828249	0.759712	0.792000
6	SVM	WoE	0.711012	0.000000	0.000000	0.000000
7	SVM	OneHot	0.711012	0.000000	0.000000	0.000000
8	Logistic Regression	WoE	0.838980	0.750074	0.663593	0.703658
9	Logistic Regression	OneHot	0.869566	0.802248	0.728031	0.763135