Task 1: Multiple Regression for Predictive Modeling

Sina Bozorgmehr

Western Governors University

Task 1: Multiple Regression for Predictive Modeling

Introduction

The fierce competition in the telecommunications industry makes it difficult to have loyal customers stay in a particular company for a long time. Customers are looking for better services with lower costs and if they are not stuck in a contract can easily switch to another company. This issue is one of the main reasons that the churn rate is normally high in the industry. As data analysts, we are trying to review the customers' data and create a model based on the trend and different metrics to predict how long a customer will stay with the company and then be prepared to try to change their mind with different actions.

Research Question

Based on the dataset, 25% of the customers left the company in the last month. Amongst them, some had been a customer only for a month and some been loyal for more than five years. It would be marvelous if the tenure, how long they will stay with the company, could be predicted based on the trend and metrics in the dataset.

In this data analysis, a multiple linear regression model will be created to predict the tenure based on a range of different predictors. Multiple features will be used to generate an initial model and with stepwise feature selection, a reduced and simpler model will be created with fewer features.

Method Justification

The Multiple Linear Regression (MLR) is used in this analysis to predict the tenure of customers before they churn. MLR makes several assumptions. Linear relationship between the dependent variable and independent variable is the first assumption. Residuals are normally distributed and there is no multicollinearity between independent variables. The last one assumes that the variance of error terms are similar across the values of independent variables that is known as homoscedasticity (*Assumptions of Multiple Linear Regression*, 2020).

In this analysis, Python and its famous packages is used for easier and faster model creation. Pandas is a python package that makes the dataset exploring very easy and fast. Also, Sci-kit learn package is the main library to create the predictive model and parameters analyses. The whole script is written and run in Jupyter Notebook for better annotation and visualization.

MLR is chosen since we have several numeric and categorical features that can be handled with such a model. Also, the dependent variable is a continuous data that MLR can interpret.

Data Preparation

The dataset has 10,000 observations with 50 variables that is checked for missing values and the type of each variable. Tenure is the dependent continuous variable, and the rest of the variables act as independent variables. To make a more precise model, only the customers who churned last month are kept for prediction of tenure (Figure 1). As mentioned in the earlier section, features include a combination of categorical and numeric variables. In order to make categorical variables with string values ready for MLR, Pandas get_dummies is used to convert this variables' values into numeric levels. Also, most of the demographic variables are remove from the dataset since they have no importance in the analysis (Figure 3). Next, categorical, and continuous variables are separated for easier visualization. Seaborn and matplotlib packages are used for univariate and bivariate visualizations. Figures 4 and 6 show the univariate visualization of numeric and categorical variables, respectively. A correlation matrix plot is created to show the possible multicollinearity between the features. As figure 5 demonstrates, Tenure and Bandwidth per year have a super strong covariation which leads to removing the Bandwidth per year from our feature set. This covariance can bee seen better in Figure 7. All eight questions in the survey demonstrate a very similar distribution in the responses and we can see this in the correlation matrix (Figure 5) as well. Therefore, all the eight questions were converted into a single variable with the code seen in figure 9. After all this preparation the final cleaned dataset was save into a csv file. This dataset contains 2,650 observations and 27 columns.

Figure 1

Codes to retrieve only the churned customers.

```
#Creating a new dataset that only have data related to churned customers
churn_df = df[df['Churn']=='Yes']
churn_df.shape
```

```
(2650, 50)
```

Figure 2

Codes to eliminate the variables related to customers demographics.

```
#eliminating variables that have no importance in our analysis

column = ['CaseOrder', 'Customer_id', 'Interaction', 'UID', 'City', 'State', 'County',
            'Zip', 'Lat', 'Lng', 'Population', 'Area', 'TimeZone','Job', 'Churn'
          ]

churn_df.drop(columns=column, inplace=True)
churn_df.head()
```

Figure 3

Creating separate continuous and categorical variables lists.

```
#Creating two lists, containing continuous and categorical variables

cont_var = ['Children', 'Age', 'Income', 'Outage_sec_perweek', 'Email', 'Contacts', 'Yearly_equip_failure',
             'Tenure', 'MonthlyCharge', 'Bandwidth_GB_Year', 'Item1', 'Item2', 'Item3', 'Item4', 'Item5', 'Item6',
             'Item7', 'Item8'
            ]
cat_var = [i for i in churn_df.columns if i not in cont_var]

print('Continuous variables are:\n\n {} \n\n and Categorical variables are:\n\n {}'.format(cont_var, cat_var))
```

Figure 4

Continuous variables visualization.

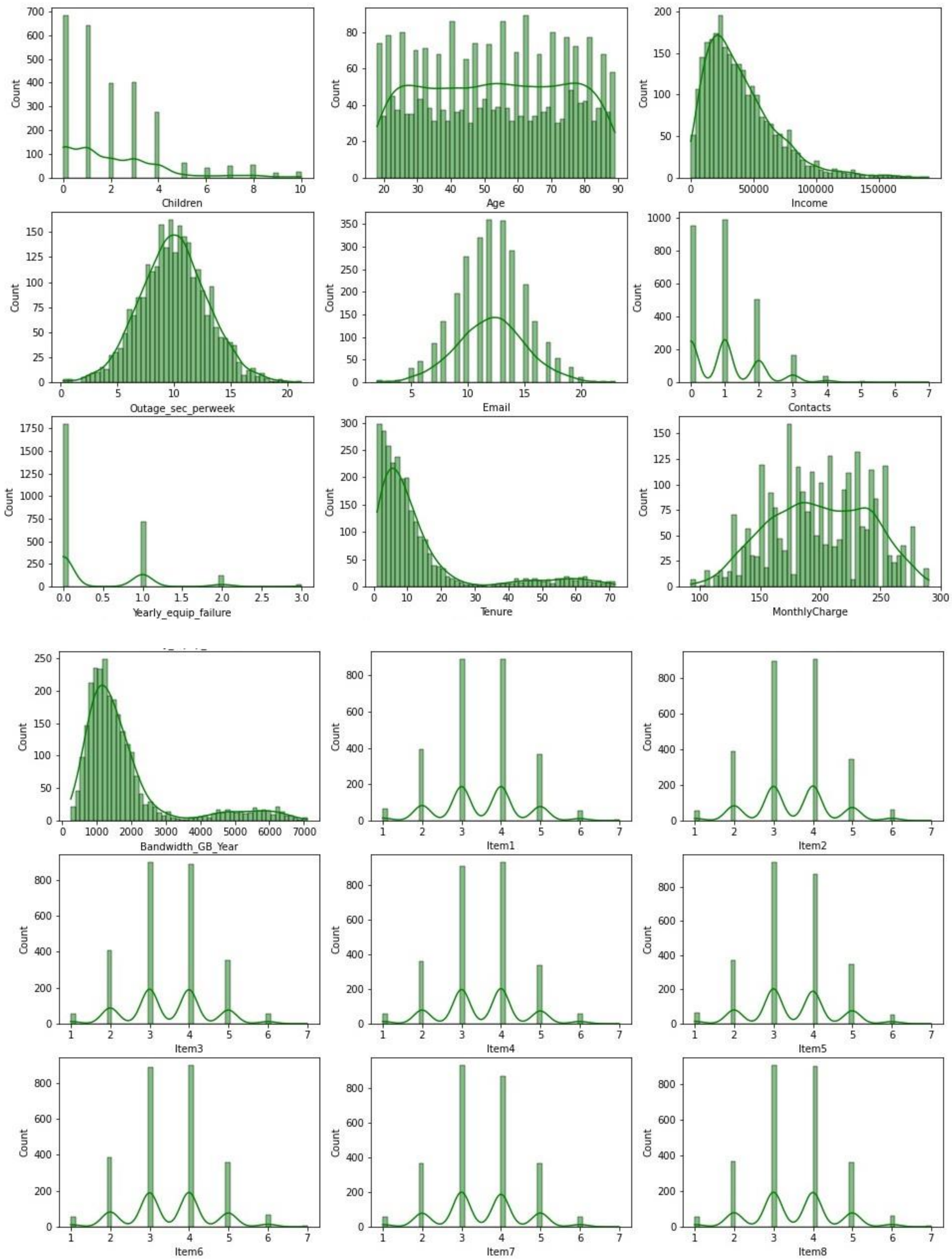


Figure 5

Correlation matrix heatmap for the continuous variables.

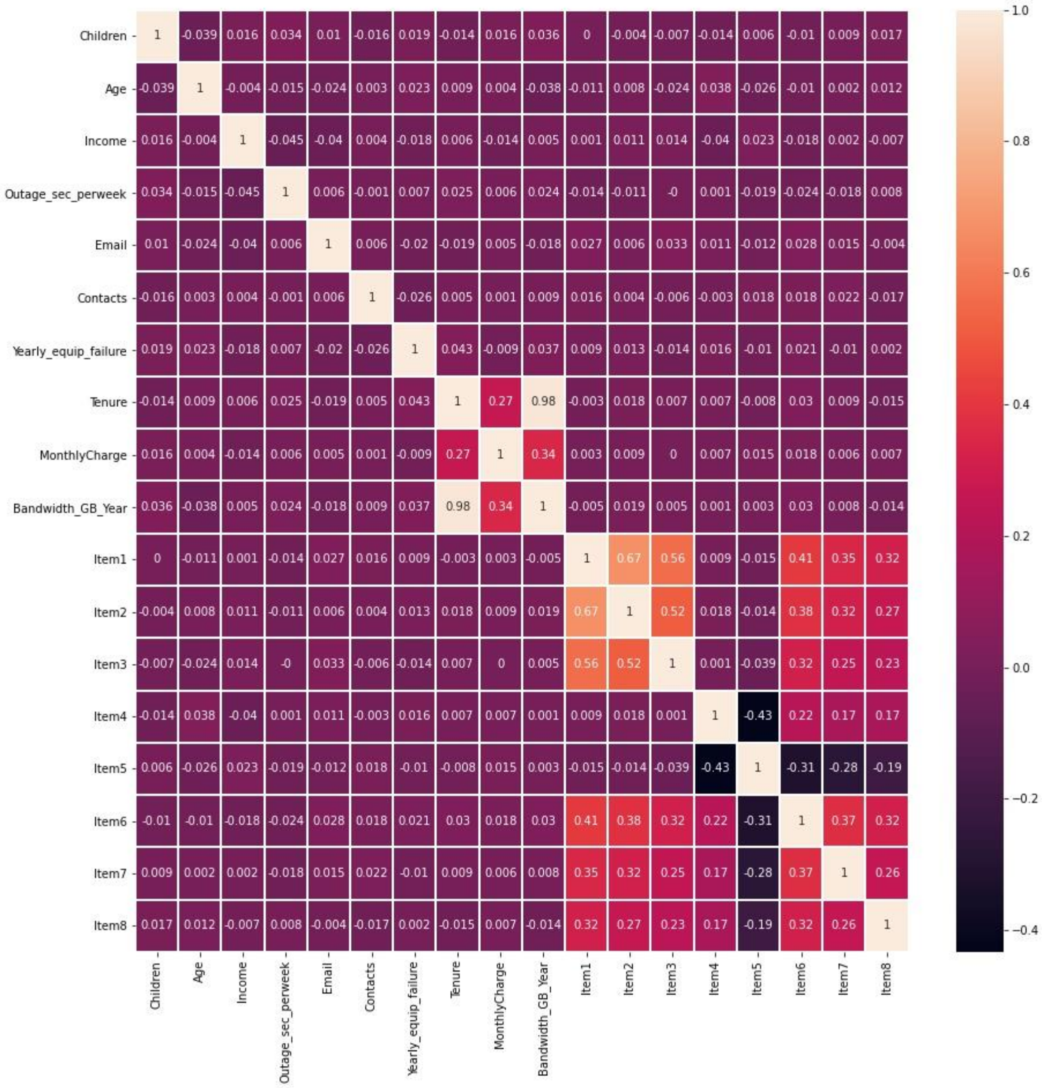


Figure 6

Categorical variables visualization.

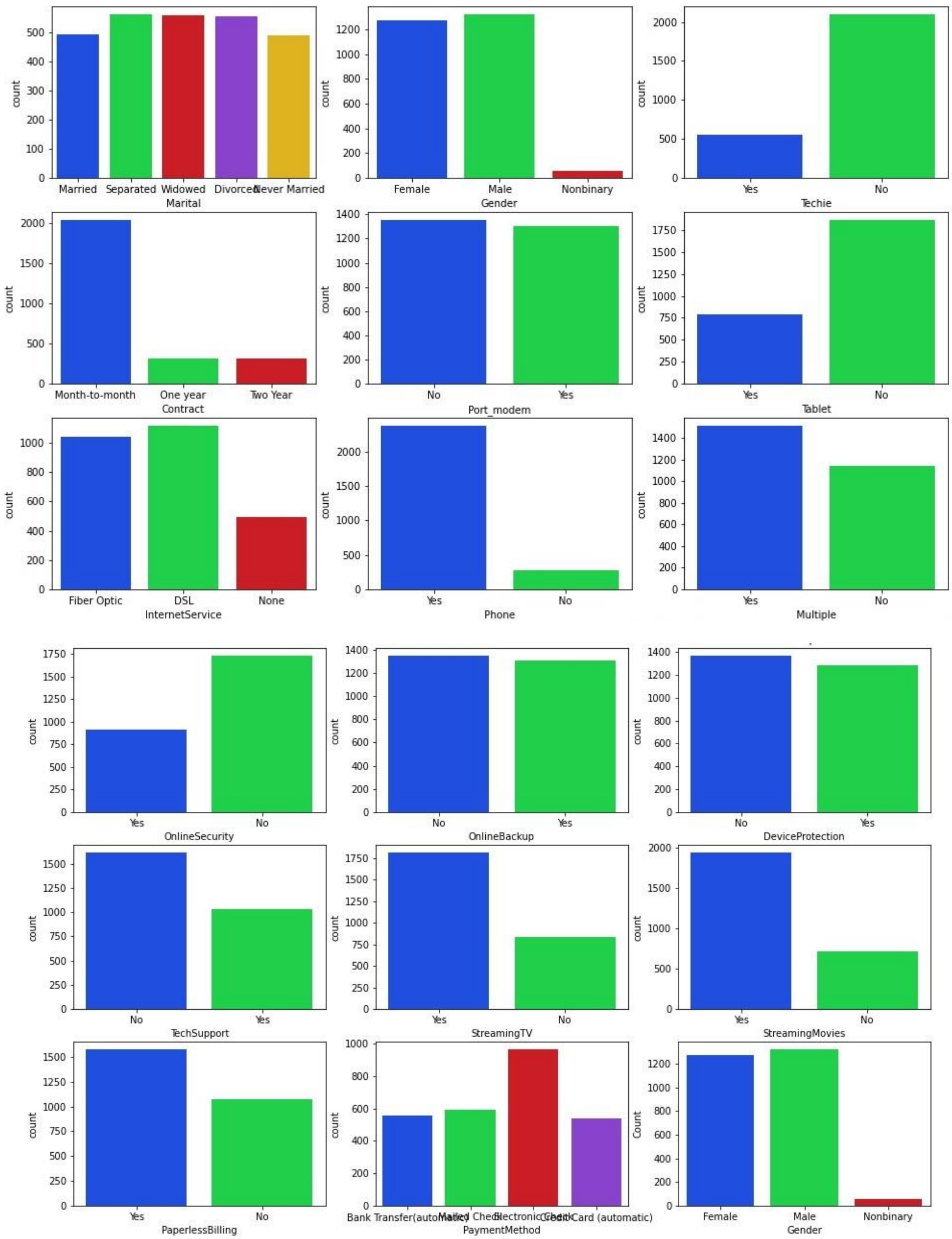


Figure 7

Bivariate visualization of continuous variables vs dependent variables.

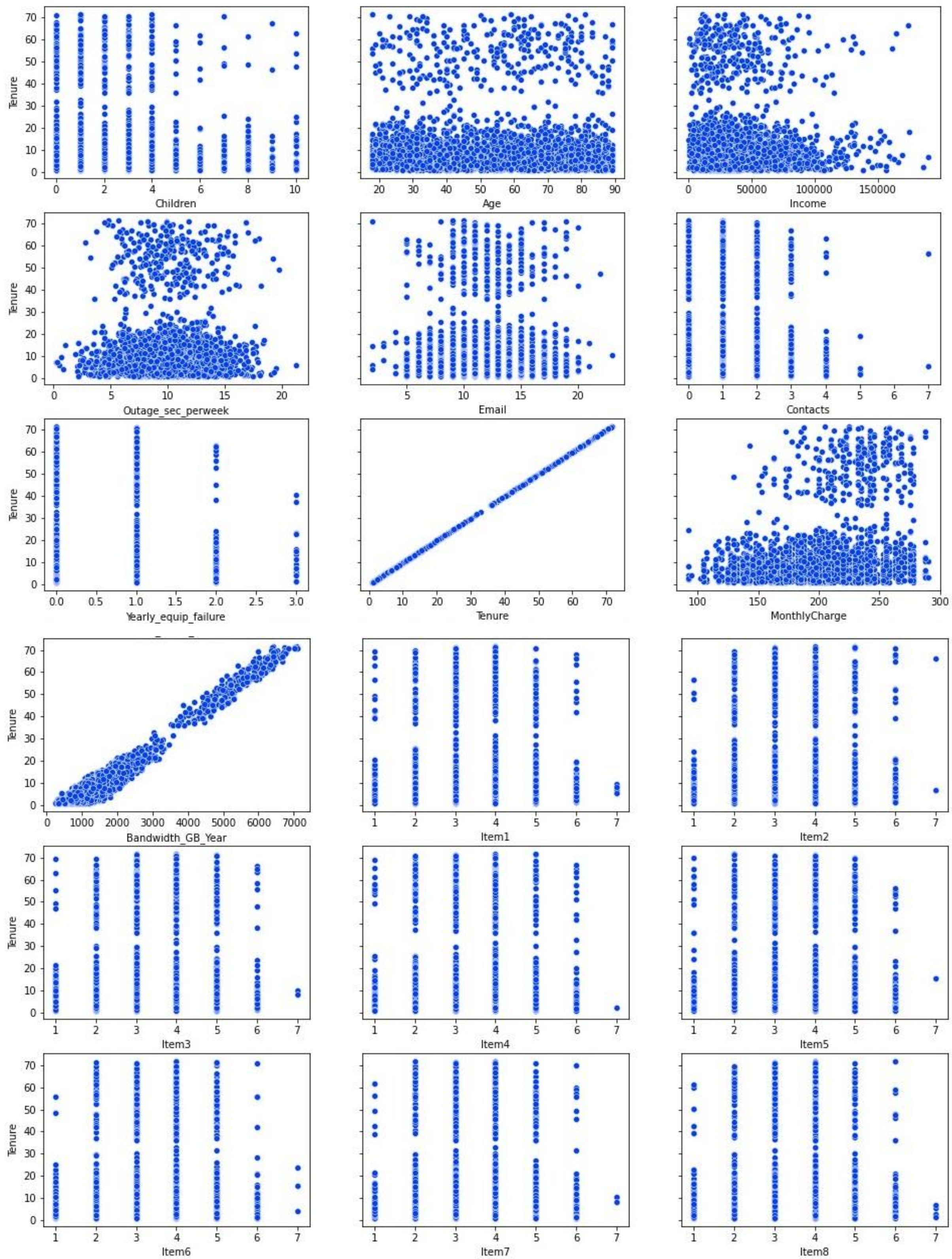
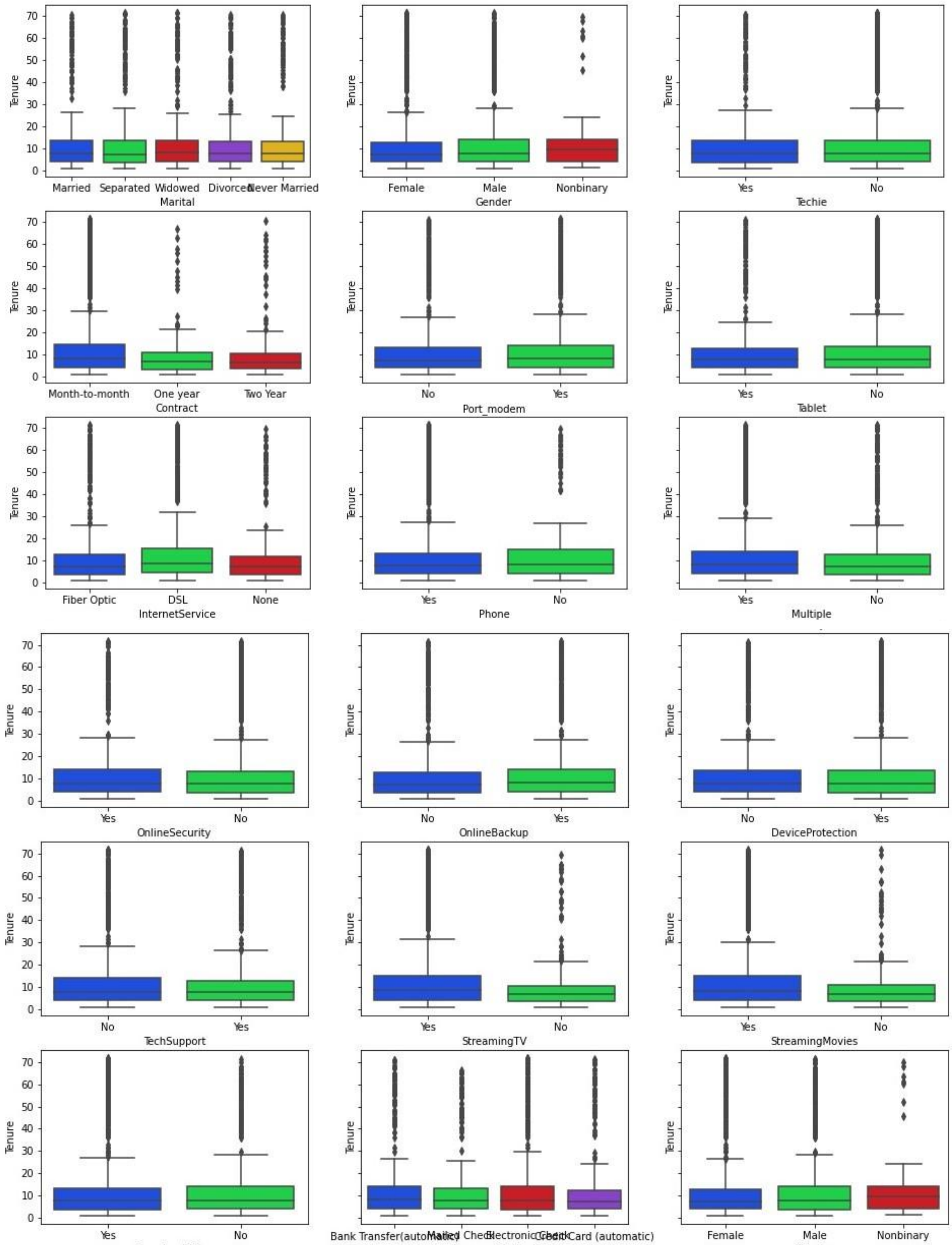


Figure 8

Bivariate visualization of categorical variables vs. dependent variable



Model Comparison and Analysis

Before starting to create the initial MLR model with all the predictors, we need to convert the categorical variables into dummy variables with numeric levels. Pandas method, get_dummies, is used to perform this function (Figure 10). One of the levels is dropped out to avoid multicollinearity (Kumar, 2020). Next, we separate the independent variables and the dependent variable and assign them to X and y, respectively. Now, the X has 34 columns including the new dummy variables.

Figure 10

Creating dummy variables

```
# creating dummy variables for all categorical features

cleaned_df = pd.get_dummies(cleaned_df, drop_first=True)
cleaned_df.head()
```

To be able to evaluate the predictive model on out of samples data, 30% of the observation is left out of training using the Sci-kit Learn method, *train_test_split* (Figure 11). 1,855 observations are dedicated for training and the remaining 795 for evaluation. The feature set includes numeric values ranging widely amongst predictors that requires the scaling before fitting the MLR model. Since the categorical variables' values are already either 1 or 0, *MinMaxSclaer* is applied to normalize all the values (Figure 12).

Figure 11

Splitting data in to training and test sets.

```
#Spliting observations into training, validation and test groups:

from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=110)


print('Training Set Observations: %.f' % X_train.shape[0])
print('Test Set Observations: %.f' % X_test.shape[0])
```

Figure 12

Normalization of datapoints.

```
# Normalizing the features

from sklearn.preprocessing import MinMaxScaler

index_train = X_train.index
index_test = X_test.index


n_scaler = MinMaxScaler()
X_train = pd.DataFrame(n_scaler.fit_transform(X_train), index=index_train, columns=X.columns)
X_test = pd.DataFrame(n_scaler.transform(X_test), index=index_test, columns=X.columns)
```

StatsModels, *OLS*, is used to train and fit the training set and create the initial MLR model (Figure 13). The summary of the initial model states that the R^2 value is 0.171 and adjusted R^2 is 0.155 with a *p-value* of almost zero (Figure 14). Looking at the coefficients' *p-values*, we notice that most of them are not significant value (Figure 15) which bolds the need for a feature selection approach to get rid of some of the predictors. Stepwise Feature Elimination is used to reduce the feature set. The forward stepwise regression method employs a series of steps that allow features to join or exit the regression model one by one. This method often uses a *p-value* threshold to determine entry and exit of the predictors to reduces the number of features (Kuhn & Johnson, 2019). In this analysis, the *p-value* threshold is set to 0.05 (Figure 16). After using this function, the independent variables are reduced to 8 predictors (Figure 17).

Figure 13

Creating the initial model.

```
#Creating the initial MLR model using Stats Models:

import statsmodels.api as sm

init_mod = sm.OLS(y_train, sm.add_constant(X_train)).fit()

init_mod.summary()
```

Figure 14

Statistics of the initial model.

OLS Regression Results

Dep. Variable:	Tenure	**R-squared:**	0.171
Model:	OLS	**Adj. R-squared:**	0.155
Method:	Least Squares	**F-statistic:**	11.02
Date:	Wed, 12 May 2021	**Prob (F-statistic):**	6.84e-53
Time:	19:38:44	**Log-Likelihood:**	-7566.9
No. Observations:	1855	**AIC:**	1.520e+04
Df Residuals:	1820	**BIC:**	1.540e+04
Df Model:	34		
Covariance Type:	nonrobust		

The reduced model is fitted based on the reduced features set using the StatsModels. The results demonstrate that the adjusted R^2 has been slightly improved. However, all the predictors have a meaningful *p-value*, now (Figure 18). Our model is less complex and still has the same power using less predictors. This model includes both categorical and continuous variables to predict the how long a customer will stay with the company.

By using the reduced model on the test group, we get an R^2 score of 0.173 (Figure 19) which is like the one from training set that demonstrates this model can work on out of sample data in the same way. A residual plot is plotted to show the distribution of residual versus the fitted values both on training (red points) and test (blue points) datapoints (Figure 20).

Figure 15

Coefficients and p-values of the initial model.

	coef	std err	t	P>|t|	[0.025	0.975]
const	5.263	2.969	1.773	0.076	-0.559	11.085
Children	1.1788	1.638	0.72	0.472	-2.033	4.391
Age	-0.8944	1.17	-0.765	0.445	-3.188	1.4
Income	-0.4387	2.228	-0.197	0.844	-4.807	3.93
Outage_sec_perweek	2.2986	2.383	0.964	0.335	-2.376	6.973
Email	-1.9667	2.231	-0.881	0.378	-6.343	2.409
Contacts	-0.1472	2.396	-0.061	0.951	-4.846	4.551
Yearly_equip_failure	2.6145	1.657	1.577	0.115	-0.636	5.865
MonthlyCharge	32.8087	9.572	3.428	0.001	14.036	51.581
Survey	0.4955	2.174	0.228	0.82	-3.768	4.759
Marital_Married	0.2435	1.057	0.23	0.818	-1.83	2.317
Marital_Never Married	0.7568	1.082	0.699	0.484	-1.366	2.879
Marital_Separated	0.3544	1.046	0.339	0.735	-1.698	2.407
Marital_Widowed	0.0002	1.044	0	1	-2.046	2.047
Gender_Male	0.1726	0.684	0.252	0.801	-1.169	1.514
Gender_Nonbinary	0.8223	2.326	0.353	0.724	-3.74	5.385
Techie_Yes	1.7143	0.826	2.075	0.038	0.094	3.335
Contract_One year	-9.4543	1.111	-8.509	0	-11.633	-7.275
Contract_Two Year	-8.3845	1.093	-7.668	0	-10.529	-6.24
Port_modem_Yes	0.597	0.675	0.884	0.377	-0.728	1.922
Tablet_Yes	-1.0412	0.736	-1.414	0.157	-2.485	0.403
InternetService_Fiber Optic	-8.3227	1.201	-6.932	0	-10.678	-5.968
InternetService_None	-3.1812	1.155	-2.755	0.006	-5.446	-0.916
Phone_Yes	-1.0832	1.075	-1.008	0.314	-3.191	1.025
Multiple_Yes	-1.8868	1.733	-1.089	0.276	-5.286	1.512
OnlineSecurity_Yes	-0.2582	0.713	-0.362	0.717	-1.656	1.14
OnlineBackup_Yes	-1.1249	1.273	-0.883	0.377	-3.622	1.372
DeviceProtection_Yes	-0.5847	0.909	-0.644	0.52	-2.367	1.197
TechSupport_Yes	-1.9614	0.908	-2.159	0.031	-3.743	-0.18
StreamingTV_Yes	0.4325	2.602	0.166	0.868	-4.671	5.536
StreamingMovies_Yes	-1.1129	3.01	-0.37	0.712	-7.016	4.791
PaperlessBilling_Yes	-0.7893	0.686	-1.151	0.25	-2.134	0.556
PaymentMethod_Credit Card (automatic)	-0.6739	1.047	-0.644	0.52	-2.727	1.379
PaymentMethod_Electronic Check	0.3204	0.931	0.344	0.731	-1.505	2.145
PaymentMethod_Mailed Check	-0.6359	1.03	-0.617	0.537	-2.656	1.384

Figure 16

Stepwise feature selection function

```
#Defining a function for Stepwise Feature Selection

def stepwise_selection(data, target,SL_in=0.05,SL_out = 0.05):
    initial_features = data.columns.tolist()
    best_features = []
    while (len(initial_features)>0):
        remaining_features = list(set(initial_features)-set(best_features))
        new_pval = pd.Series(index=remaining_features, dtype='float64')
        for new_column in remaining_features:
            model = sm.OLS(target, sm.add_constant(data[best_features+[new_column]])).fit()
            new_pval[new_column] = model.pvalues[new_column]
        min_p_value = new_pval.min()
        if(min_p_value<SL_in):
            best_features.append(new_pval.idxmin())
            while(len(best_features)>0):
                best_features_with_constant = sm.add_constant(data[best_features])
                p_values = sm.OLS(target, best_features_with_constant).fit().pvalues[1:]
                max_p_value = p_values.max()
                if(max_p_value >= SL_out):
                    excluded_feature = p_values.idxmax()
                    best_features.remove(excluded_feature)
                else:
                    break
        else:
            break
    return best_features
```

Figure 17

Selected feature using the stepwise approach

```
#Selected Features

features =stepwise_selection(X_train, y_train)
features
```

```
['MonthlyCharge',
 'InternetService_Fiber Optic',
 'Contract_One year',
 'Contract_Two Year',
 'InternetService_None',
 'StreamingTV_Yes',
 'TechSupport_Yes',
 'Techie_Yes']
```

Figure 18

Statistics and coefficient results for the reduced model

OLS Regression Results

Dep. Variable:	Tenure	**R-squared:**	0.163
Model:	OLS	**Adj. R-squared:**	0.159
Method:	Least Squares	**F-statistic:**	44.96
Date:	Wed, 12 May 2021	**Prob (F-statistic):**	2.54e-66
Time:	19:56:12	**Log-Likelihood:**	-7575.5
No. Observations:	1855	**AIC:**	1.517e+04
Df Residuals:	1846	**BIC:**	1.522e+04
Df Model:	8		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P>|t|**	**[0.025**	**0.975]**
const	3.2648	1.042	3.134	0.002	1.222	5.308
MonthlyCharge	26.7875	2.085	12.849	0.000	22.699	30.876
InternetService_Fiber Optic	-7.7356	0.776	-9.965	0.000	-9.258	-6.213
Contract_One year	-9.4238	1.101	-8.562	0.000	-11.582	-7.265
Contract_Two Year	-8.1505	1.081	-7.542	0.000	-10.270	-6.031
InternetService_None	-3.6348	0.948	-3.834	0.000	-5.494	-1.776
StreamingTV_Yes	2.1265	0.857	2.482	0.013	0.446	3.807
TechSupport_Yes	-1.5448	0.694	-2.226	0.026	-2.906	-0.184
Techie_Yes	1.6578	0.818	2.028	0.043	0.054	3.261

Omnibus:	565.123	**Durbin-Watson:**	2.078
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	1425.639
Skew:	1.649	**Prob(JB):**	2.67e-310
Kurtosis:	5.751	**Cond. No.**	10.4

Figure 19

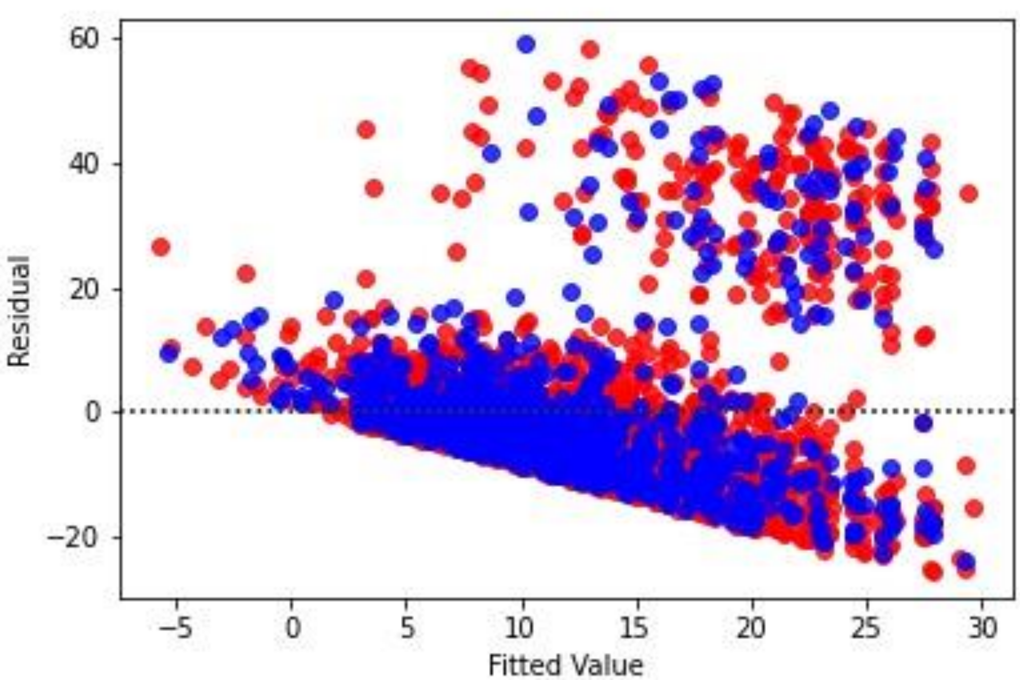
R^2 score of the reduced model for both training and test sets.

```
#Printing R2 of the reduced model

from sklearn.metrics import r2_score

print('R2 for the training set is %.3f' % r2_score(y_train, y_pred))
print('R2 for the test set is %.3f' % r2_score(y_test, y_pred_test))
```

```
R2 for the training set is 0.163
R2 for the test set is 0.173
```

Figure 20

Residual plot of the reduced model



Data Summary and Implications

Figure 21 shows the regression equation resulted from the reduced model that can be used to predict the tenure of customers. Monthly charge has the largest positive coefficient amongst the predictors. Although it might seem that the more money the customer pays per months the longer the tenure, but it is not that simple. It might be the result of more adds on that customer has. Both One-Year-Contract and Two-Year-Contract have negative coefficients that suggest being in a contract lowers the tenure which is in contradiction with the data visualizations. One reason could be hidden in the monthly charge as we mentioned earlier. Having no internet service has a negative effect on tenure and so dose the Tech-Support. On the other hand, being a techie and streaming TV adds-on have a positive effect.

Figure 21

Regression equation of the reduced model

```
     Tenure = 3.26 + 26.79 MonthlyCharge + -7.74 InternetService_Fiber Optic +
-9.42 Contract_One year + -8.15 Contract_Two Year + -3.63 InternetService_None +
2.13 StreamingTV_Yes +-1.54 TechSupport_Yes + 1.66 Techie_Yes
```

The statistics shows that only about 16% of the variation in the data can be interpreted by this MLR model. Telecommunication industry is a complicated field that has multiple factors, and the results suggest that may be MLR is not a practical model for predicting the customers' tenure.

However, based on this result and accepting the limitations, the company can predict which customers might be churning during the next month and what are the main reasons for their decisions. Defining those, the marketing team could come up with different ideas to prevent some of the potential churning customers from doing so.

Sources

The dataset used in this analysis was acquired form WGU portal:

https://access.wgu.edu/ASP3/aap/content/d9rkejv84kd9rk30fi2l.zip

The stepwise feature selection function was acquired form:

https://www.analyticsvidhya.com/blog/2020/10/a-comprehensive-guide-to-feature-selection-using-wrapper-methods-in-python/

References

Assumptions of Multiple Linear Regression. (2020, March 10). Statistics Solutions. https://www.statisticssolutions.com/assumptions-of-multiple-linear-regression/

Kuhn, M., & Johnson, K. (2019). *Feature Engineering and Selection: A Practical Approach for Predictive Models* (1st ed.). Chapman and Hall/CRC.

Kumar, S. (2020, December 27). *How to avoid multicollinearity in Categorical Data - Towards Data Science*. Medium. https://towardsdatascience.com/how-to-avoid-multicollinearity-in-categorical-data-46eb39d9cd0d