In [280]:

```
# Import libraries
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import statsmodels.api as sm
import numpy as np
from sklearn.metrics import accuracy_score, confusion_matrix

# Import dataset
df = pd.read_csv("C:/Users/hkeim/OneDrive/Documents/School/D208/churn_clean.csv")
```

In [281]:

```
# Create dummies for bianry objects
df=pd.get_dummies(df, columns=['Churn', 'Techie', 'Port_modem', 'Tablet', 'Phone', 'Multiple'
                                            'OnlineSecurity', 'OnlineBackup', 'DeviceProte
```

In [282]:

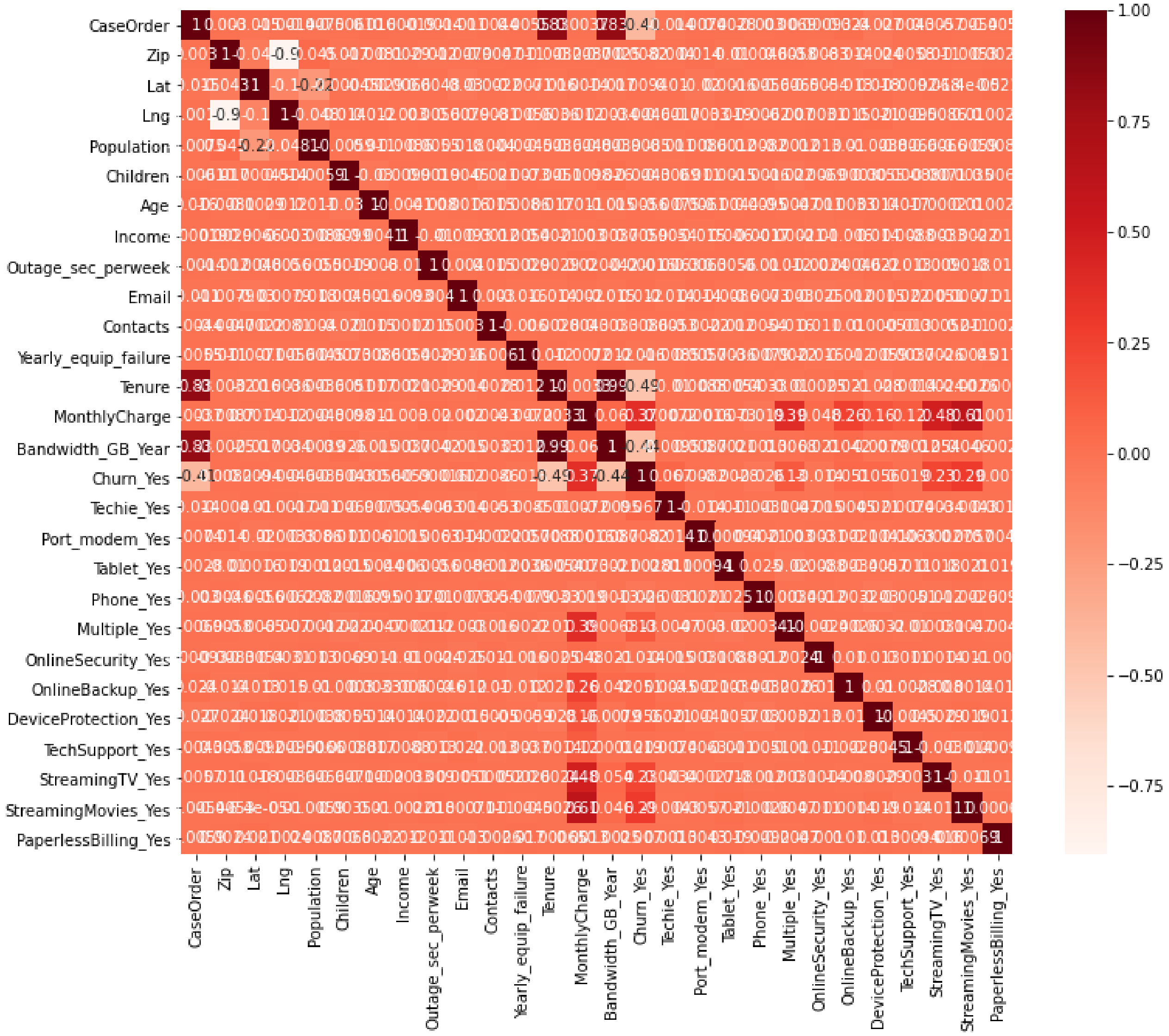
```
# Drop 'No' dummies
df=df.drop(['Churn_No', 'Techie_No', 'Port_modem_No', 'Tablet_No', 'Phone_No', 'Multiple_No',
           'StreamingTV_No', 'StreamingMovies_No', 'PaperlessBilling_No'], axis=1)
```

In [283]:

```
#drop unnecessary variables
df=df.drop(['Customer_id', 'Interaction', 'UID', 'City', 'County', 'Job', 'Item1', 'Item2', '
                'Item5', 'Item6', 'Item7', 'Item8'], axis=1)
```

In [284]:

```
# Use Pearson Correlation to choose initial model variables
plt.figure(figsize=(12,10))
cor = df.corr()
sns.heatmap(cor, annot=True, cmap=plt.cm.Reds)
plt.show()
```



In [285]:

```
# Create dataframe with initial variables
df=df.filter(items=['CaseOrder', 'Tenure', 'MonthlyCharge', 'Bandwidth_GB_Year', 'Churn_Yes',
```

In [286]:

```
# Summary statistics
print(df.describe())
```

```
         CaseOrder        Tenure  MonthlyCharge  Bandwidth_GB_Year  \
count  10000.00000  10000.000000   10000.000000       10000.000000   
mean    5000.50000     34.526188     172.624816        3392.341550   
std     2886.89568     26.443063      42.943094        2185.294852   
min        1.00000      1.000259      79.978860         155.506715   
25%     2500.75000      7.917694     139.979239        1236.470827   
50%     5000.50000     35.430507     167.484700        3279.536903   
75%     7500.25000     61.479795     200.734725        5586.141370   
max    10000.00000     71.999280     290.160419        7158.981530   

          Churn_Yes  Multiple_Yes  StreamingTV_Yes  StreamingMovies_Yes  
count  10000.000000  10000.000000     10000.000000         10000.000000  
mean       0.265000      0.460800         0.492900             0.489000  
std        0.441355      0.498486         0.499975             0.499904  
min        0.000000      0.000000         0.000000             0.000000  
25%        0.000000      0.000000         0.000000             0.000000  
50%        0.000000      0.000000         0.000000             0.000000  
```

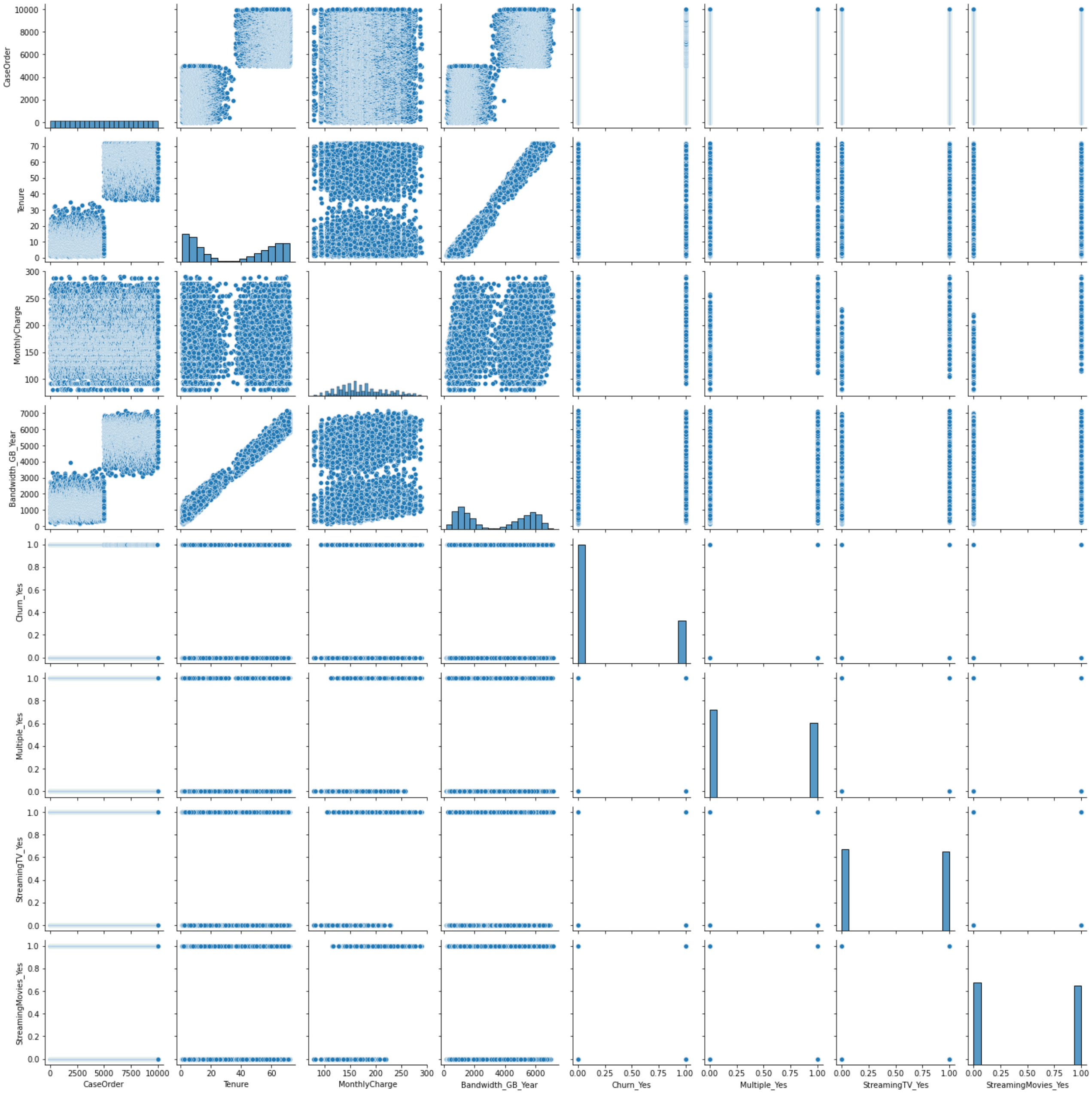
```
75%        1.000000      1.000000         1.000000            1.000000
max        1.000000      1.000000         1.000000            1.000000
```

In [287]:

```
# Univariate and bivariate visualizations
sns.pairplot(df)
```

Out[287]:

```
<seaborn.axisgrid.PairGrid at 0x2a7451fc6d0>
```



In [288]:

```
# Defining the independent variables
X=df[['CaseOrder', 'Tenure', 'MonthlyCharge', 'Bandwidth_GB_Year', 'Multiple_Yes', 'Streaming
```

In [289]:

```
# Get initial model intercept
X = sm.add_constant(X)
```

In [290]:

```
# Defining the dependent Variable
y=df['Churn_Yes']
```

In [291]:

```
# Initial logistic regression model
model = sm.Logit(endog=y, exog=X).fit()
print(model.summary())
```

```
Optimization terminated successfully.
         Current function value: inf
         Iterations 8
                           Logit Regression Results
==============================================================================
Dep. Variable:              Churn_Yes   No. Observations:                10000
Model:                          Logit   Df Residuals:                     9992
Method:                           MLE   Df Model:                            7
Date:                Sat, 09 Oct 2021   Pseudo R-squ.:                     inf
Time:                        21:46:55   Log-Likelihood:                   -inf
converged:                       True   LL-Null:                        0.0000
Covariance Type:            nonrobust   LLR p-value:                     1.000
=======================================================================================
                          coef    std err          z      P>|z|      [0.025      0.975]
---------------------------------------------------------------------------------------
const                  -3.8150      0.193    -19.784      0.000      -4.193      -3.437
CaseOrder            6.086e-06   2.01e-05      0.302      0.762   -3.34e-05    4.55e-05
Tenure                 -0.2347      0.012    -19.684      0.000      -0.258      -0.211
MonthlyCharge           0.0104      0.002      6.730      0.000       0.007       0.013
Bandwidth_GB_Year       0.0019      0.000     13.694      0.000       0.002       0.002
Multiple_Yes            0.6336      0.082      7.766      0.000       0.474       0.793
StreamingTV_Yes         1.1356      0.099     11.498      0.000       0.942       1.329
StreamingMovies_Yes     1.4812      0.109     13.642      0.000       1.268       1.694
=======================================================================================
```

In [292]:

```
# Initial model odds ratios
np.exp(model.params)
```

Out[292]:

```
const                  0.022037
CaseOrder              1.000006
Tenure                 0.790815
MonthlyCharge          1.010412
Bandwidth_GB_Year      1.001879
Multiple_Yes           1.884349
StreamingTV_Yes        3.113014
StreamingMovies_Yes    4.398402
dtype: float64
```

In [293]:

```
# Initial model predictions
pred = model.predict(exog=X)
pred.head()
```

Out[293]:

```
0    0.390642
1    0.960092
2    0.528901
3    0.200313
4    0.267200
dtype: float64
```

In [294]:

```
# Initial model rounded predictions
round(pred)
```

Out[294]:

```
0       0.0
1       1.0
2       1.0
3       0.0
4       0.0
       ...
```

```
9995    0.0
9996    0.0
9997    0.0
9998    0.0
9999    0.0
Length: 10000, dtype: float64
```

In [295]:

```
# Initial model confusion matrix
confusion_matrix(y_true=list(y), y_pred=list(round(pred)))
```

Out[295]:

```
array([[6799,  551],
       [ 944, 1706]], dtype=int64)
```

In [296]:

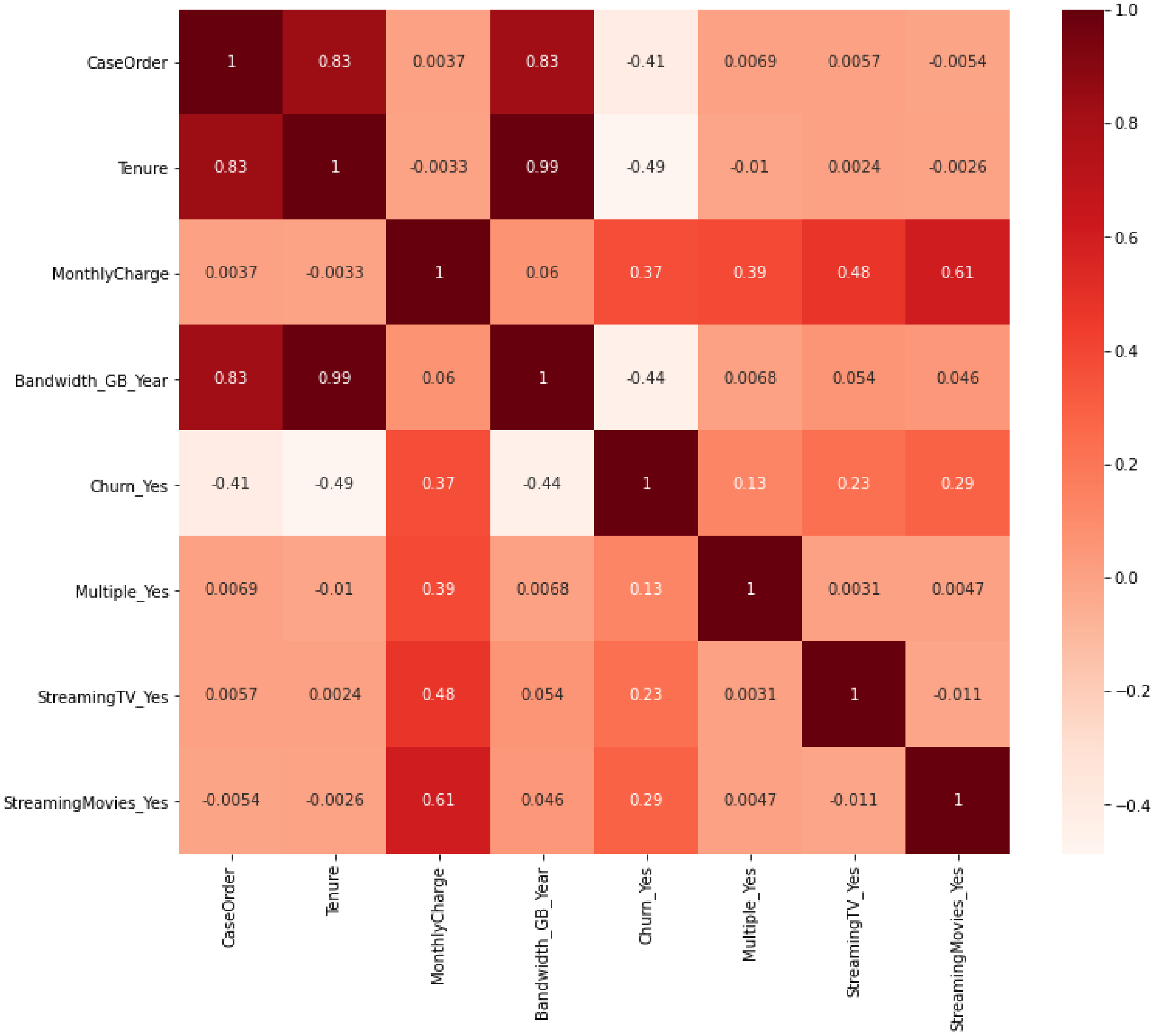
```
# Initial model accuracy of the fitted model
accuracy_score(y_true=list(y), y_pred=list(round(pred)))
```

Out[296]:

```
0.8505
```

In [297]:

```
# Use Pearson Correlation to choose reduced model variables
plt.figure(figsize=(12,10))
cor = df.corr()
sns.heatmap(cor, annot=True, cmap=plt.cm.Reds)
plt.show()
```



In [298]:

```
# Pairwise correlation analysis
X.corr()
```

Out[298]:

	const	CaseOrder	Tenure	MonthlyCharge	Bandwidth_GB_Year	Multiple_Yes	Streamin
const	NaN	NaN	NaN	NaN	NaN	NaN	
CaseOrder	NaN	1.000000	0.832550	0.003677	0.825561	0.006915	
Tenure	NaN	0.832550	1.000000	-0.003337	0.991495	-0.010422	
MonthlyCharge	NaN	0.003677	-0.003337	1.000000	0.060406	0.385979	
Bandwidth_GB_Year	NaN	0.825561	0.991495	0.060406	1.000000	0.006823	
Multiple_Yes	NaN	0.006915	-0.010422	0.385979	0.006823	1.000000	
StreamingTV_Yes	NaN	0.005690	0.002440	0.482312	0.054314	0.003097	
StreamingMovies_Yes	NaN	-0.005353	-0.002574	0.608115	0.045600	0.004691	-

In [299]:

```
# Defining the reduced independent variables
Xr=df[['MonthlyCharge', 'Bandwidth_GB_Year', 'Multiple_Yes', 'StreamingTV_Yes']]
```

In [300]:

```
# Get reduced model intercept
Xr = sm.add_constant(Xr)
```

In [301]:

```
# Reduced logistic regression model
modelr = sm.Logit(endog=y, exog=Xr).fit()
print(modelr.summary())
```

```
Optimization terminated successfully.
         Current function value: inf
         Iterations 7
                           Logit Regression Results
==============================================================================
Dep. Variable:              Churn_Yes   No. Observations:                10000
Model:                          Logit   Df Residuals:                     9995
Method:                           MLE   Df Model:                            4
Date:                Sat, 09 Oct 2021   Pseudo R-squ.:                     inf
Time:                        21:48:25   Log-Likelihood:                   -inf
converged:                       True   LL-Null:                        0.0000
Covariance Type:            nonrobust   LLR p-value:                     1.000
=====================================================================================
                        coef    std err          z      P>|z|      [0.025      0.975]
-------------------------------------------------------------------------------------
const                -4.7256      0.149    -31.783      0.000      -5.017      -4.434
MonthlyCharge         0.0311      0.001     30.962      0.000       0.029       0.033
Bandwidth_GB_Year    -0.0008      2e-05    -41.530      0.000      -0.001      -0.001
Multiple_Yes          0.0184      0.065      0.284      0.777      -0.109       0.145
StreamingTV_Yes       0.5945      0.069      8.635      0.000       0.460       0.729
=====================================================================================
```

In [302]:

```
# Reduced model odds ratios
np.exp(modelr.params)
```

Out[302]:

```
const                0.008866
MonthlyCharge        1.031545
Bandwidth_GB_Year    0.999171
Multiple_Yes         1.018540
StreamingTV_Yes      1.812107
dtype: float64
```

In [303]:

```
# Reduced model predictions
pred = modelr.predict(exog=Xr)
pred.head()
```

Out[303]:

```
0    0.470017
1    0.940376
2    0.190846
3    0.099609
4    0.574606
dtype: float64
```

In [304]:

```
# Reduced model rounded predictions
round(predr)
```

Out[304]:

```
0       0.0
1       1.0
2       0.0
3       0.0
4       1.0
       ...
9995    0.0
```

```
9996    0.0
9997    0.0
9998    0.0
9999    0.0
Length: 10000, dtype: float64
```

In [305]:

```
# Reduced model confusion matrix
confusion_matrix(y_true=list(y), y_pred=list(round(predr)))
```

Out[305]:

```
array([[6744,  606],
       [1080, 1570]], dtype=int64)
```

In [306]:

```
# Reduced model accuracy of the fitted model
accuracy_score(y_true=list(y), y_pred=list(round(predr)))
```

Out[306]:

```
0.8314
```

In [307]:

```
# Export regression data set
df.to_csv("C:/Users/hkeim/OneDrive/Documents/School/D208/Keim D208 Task Two Prepared Data.csv
```

In [ ]: