```
/*With 21 predictor variables we try to predict whether a particular customer will switch to another telecom provider or not.

/*Step 1: Importing and Merging Data*/

/*Importing Churn_Data File*/
%web_drop_table(WORK.IMPORT);
FILENAME REFFILE '/home/u63664393/Churn_Data.xlsx';
PROC IMPORT DATAFILE=REFFILE
    DBMS=XLSX
    OUT=WORK.IMPORT;
    GETNAMES=YES;
RUN;
PROC CONTENTS DATA=WORK.IMPORT; RUN;
%web_open_table(WORK.IMPORT);


/*Importing Customer_Data File*/
%web_drop_table(WORK.IMPORT1);
FILENAME REFFILE '/home/u63664393/Customer_Data.xlsx';
PROC IMPORT DATAFILE=REFFILE
    DBMS=XLSX
    OUT=WORK.IMPORT1;
    GETNAMES=YES;
RUN;
PROC CONTENTS DATA=WORK.IMPORT1; RUN;
%web_open_table(WORK.IMPORT1);


/*Importing Internet_Data File*/
%web_drop_table(WORK.IMPORT2);
FILENAME REFFILE '/home/u63664393/Internet_Data.xlsx';
PROC IMPORT DATAFILE=REFFILE
    DBMS=XLSX
    OUT=WORK.IMPORT2;
    GETNAMES=YES;
RUN;
PROC CONTENTS DATA=WORK.IMPORT2; RUN;
%web_open_table(WORK.IMPORT2);



/* Sort the datasets by the key variable 'CustomerID' */
proc sort data=WORK.IMPORT;
    by CustomerID;
run;
proc sort data=WORK.IMPORT1;
    by CustomerID;
run;
proc sort data=WORK.IMPORT2;
    by CustomerID;
run;

/* Merge the datasets */
data WORK.CombinedData;
    merge WORK.IMPORT (in=a)
          WORK.IMPORT1 (in=b)
          WORK.IMPORT2 (in=c);
    by CustomerID;
    if a and b and c; /* This line ensures that only observations present in all datasets are included */
run;

/* View the contents of the combined dataset to verify */
proc contents data=WORK.CombinedData;
run;

%web_open_table(WORK.CombinedData);




/*Step 2: Inspecting the Dataframe*/

proc print data=WORK.CombinedData(obs=10);
run;

/* Checking the dimensions of the CombinedData dataframe */
```

```
proc sql;
    select count(*) as NumberOfRows, /* Counting the number of rows */
           (select count(*) /* Counting the number of columns */
            from dictionary.columns
            where libname='WORK' and memname='COMBINEDDATA') as NumberOfColumns
    from WORK.CombinedData;
quit;

/* Getting statistical summary of numerical variables of the CombinedData dataframe */
proc means data=WORK.CombinedData N mean std min P25 median P50 P75 max;
run;

/* Show the type of each column in the CombinedData dataset */
proc contents data=WORK.CombinedData varnum;
run;




/*Step 3: Data Preparation*/

/* Data step to convert binary variables 'Yes'/'No' to 1/0 in place */
data WORK.CombinedData;
    set WORK.CombinedData; /* Read in the original dataset */

    /* Convert 'Yes'/'No' to 1/0 for each variable directly */
    if PhoneService = 'Yes' then PhoneService = '1';
    else if PhoneService = 'No' then PhoneService = '0';

    if PaperlessBilling = 'Yes' then PaperlessBilling = '1';
    else if PaperlessBilling = 'No' then PaperlessBilling = '0';

    if Churn = 'Yes' then Churn = '1';
    else if Churn = 'No' then Churn = '0';

    if Partner = 'Yes' then Partner = '1';
    else if Partner = 'No' then Partner = '0';

    if Dependents = 'Yes' then Dependents = '1';
    else if Dependents = 'No' then Dependents = '0';

    /* Convert the character variables to numeric */
    PhoneService = input(PhoneService, best.);
    PaperlessBilling = input(PaperlessBilling, best.);
    Churn = input(Churn, best.);
    Partner = input(Partner, best.);
    Dependents = input(Dependents, best.);
run;

/* Show the first few rows of the dataset to verify changes */
proc print data=WORK.CombinedData(obs=5);
run;


/* Data step to convert binary variables 'Male'/'Female' to 1/0 in place */
data WORK.CombinedData;
    set WORK.CombinedData;

    /* Convert 'Male'/'Female' to 1/0 for each variable directly */
    if gender = 'Male' then gender_male = '1';
    else if gender = 'Female' then gender_male = '0';

    /* Convert the character variables to numeric */
    gender_male = input(gender_male, best.);
run;
proc print data=WORK.CombinedData(obs=5);
run;

/* Data step to convert Contract into Binary Variable 'One year'/'Two year' to 1/0 in place, else =0 (Month-to-month) */
data CombinedData;
    set CombinedData;

    /* Step 2: Create separate binary variables for each level */
    if Contract = 'One year' then Contract_OneYear = 1; else Contract_OneYear = 0;
    if Contract = 'Two year' then Contract_TwoYear = 1; else Contract_TwoYear = 0;
run;
proc print data=WORK.CombinedData(obs=15);
```

```
run;

/* Create dummy variables for the 'PaymentMethod' variable in dataset CombinedData */
data CombinedData;
    set CombinedData;

    if PaymentMethod = 'Credit card (automatic)' then PaymentMethod_CreditCard = 1; else PaymentMethod_CreditCard = 0;
    if PaymentMethod = 'Electronic check' then PaymentMethod_ElectronicCheck = 1; else PaymentMethod_ElectronicCheck = 0;
    if PaymentMethod = 'Mailed check' then PaymentMethod_MailedCheck = 1; else PaymentMethod_MailedCheck = 0;
run;

proc print data=CombinedData(obs=15);
run;

/* Create dummy variables for the 'InternetService' variable in dataset CombinedData */
data CombinedData;
    set CombinedData;

    if InternetService = 'Fiber optic' then InternetService_FiberOptic = 1; else InternetService_FiberOptic = 0;
    if InternetService = 'No' then InternetService_No = 1; else InternetService_No = 0;
run;

proc print data=CombinedData(obs=15);
run;

/* Create dummy variables for the 'MultipleLines' variable in dataset CombinedData */
data CombinedData;
    set CombinedData;

    if MultipleLines = 'No' then MultipleLines_No = 1; else MultipleLines_No = 0;
    if MultipleLines = 'Yes' then MultipleLines_Yes = 1; else MultipleLines_Yes = 0;
run;

/* Print the dataset to view the new variables */
proc print data=CombinedData(obs=15);
run;

/* Create dummy variables for the 'OnlineSecurity' variable in dataset CombinedData */
data CombinedData;
    set CombinedData;

    if OnlineSecurity = 'No' then OnlineSecurity_No = 1; else OnlineSecurity_No = 0;
    if OnlineSecurity = 'Yes' then OnlineSecurity_Yes = 1; else OnlineSecurity_Yes = 0;
run;

proc print data=CombinedData(obs=25);
run;

/* Create dummy variables for the 'OnlineBackup' variable in dataset CombinedData */
data CombinedData;
    set CombinedData;

    if OnlineBackup = 'No' then OnlineBackup_No = 1; else OnlineBackup_No = 0;
    if OnlineBackup = 'Yes' then OnlineBackup_Yes = 1; else OnlineBackup_Yes = 0;
run;

proc print data=CombinedData(obs=25);
run;

/* Create dummy variables for the 'DeviceProtection' variable in dataset CombinedData */
data CombinedData;
    set CombinedData;

    if DeviceProtection = 'No' then DeviceProtection_No = 1; else DeviceProtection_No = 0;
    if DeviceProtection = 'Yes' then DeviceProtection_Yes = 1; else DeviceProtection_Yes = 0;
run;

proc print data=CombinedData(obs=25);
run;

/* Create dummy variables for the 'TechSupport' variable in dataset CombinedData */
data CombinedData;
    set CombinedData;

    if TechSupport = 'No' then TechSupport_No = 1; else TechSupport_No = 0;
    if TechSupport = 'Yes' then TechSupport_Yes = 1; else TechSupport_Yes = 0;
run;
```

```
proc print data=CombinedData(obs=25);
run;

/* Create dummy variables for the 'StreamingTV' variable in dataset CombinedData */
data CombinedData;
    set CombinedData;

    if StreamingTV = 'No' then StreamingTV_No = 1; else StreamingTV_No = 0;
    if StreamingTV = 'Yes' then StreamingTV_Yes = 1; else StreamingTV_Yes = 0;
run;

proc print data=CombinedData(obs=25);
run;

/* Create dummy variables for the 'StreamingMovies' variable in dataset CombinedData */
data CombinedData;
    set CombinedData;

    if StreamingMovies = 'No' then StreamingMovies_No = 1; else StreamingMovies_No = 0;
    if StreamingMovies = 'Yes' then StreamingMovies_Yes = 1; else StreamingMovies_Yes = 0;
run;
proc print data=CombinedData(obs=25);
run;

/*Dropping the repeated variables*/
data CombinedData;
    set CombinedData;
    drop Contract PaymentMethod gender StreamingTV StreamingMovies OnlineSecurity OnlineBackup DeviceProtection TechSupport Mu
run;
proc print data=CombinedData(obs=25);
run;

/* Show the type of each column in the updated dataset */
proc contents data=WORK.CombinedData varnum;
run;

/*The variable was imported as a string we need to convert it to float*/
data CombinedData;
    set CombinedData;

    TotalCharges_numeric= input(TotalCharges, best12.);
    drop TotalCharges;
run;


data CombinedData;
    set CombinedData(rename=(TotalCharges_numeric=TotalCharges));
run;

data WORK.CombinedData;
    set WORK.CombinedData;

    /* Convert character variables to float using INPUT function */
    PhoneService_numeric = input(PhoneService, 8.);
    PaperlessBilling_numeric = input(PaperlessBilling, 8.);
    Churn_numeric = input(Churn, 8.);
    Partner_numeric = input(Partner, 8.);
    Dependents_numeric = input(Dependents, 8.);
    gender_male_numeric = input(gender_male, 8.);
    drop PhoneService PaperlessBilling Churn Partner Dependents gender_male;

run;

data WORK.CombinedData;
    set WORK.CombinedData;

    rename PhoneService_numeric = PhoneService
           PaperlessBilling_numeric = PaperlessBilling
           Churn_numeric = Churn
           Partner_numeric = Partner
           Dependents_numeric = Dependents
           gender_male_numeric = gender_male;
run;
```

```
/* Show the type of each column in the updated dataset */
proc contents data=WORK.CombinedData varnum;
run;


/* Generate summary statistics for selected variables */
proc means data=WORK.CombinedData n mean std min p25 median p50 p75 max;
    var tenure MonthlyCharges SeniorCitizen TotalCharges;
run;



/*Calculate the number of missing values for each variable */
proc means data=WORK.CombinedData nmiss;
    output out=MissingCounts (drop=_type_ _freq_)
           nmiss=Num_Null_Values;
run;
proc print data=MissingCounts noobs;
run;

/* Create a filtered dataset without NaN values in TotalCharges */
data WORK.CombinedData_filtered;
    set WORK.CombinedData;
    where not missing(TotalCharges);
run;

/*Calculate the number of missing values for each variable in filtered dataset*/
proc means data=WORK.CombinedData_filtered nmiss;
    output out=MissingCounts_filtered (drop=_type_ _freq_)
           nmiss=Num_Null_Values;
run;
proc print data=MissingCounts_filtered noobs;
run;




/*Step 4: Correlation*/
proc corr data=WORK.CombinedData_filtered noprint outp=WORK.CorrelationMatrixFiltered;
run;

proc print data=WORK.CorrelationMatrixFiltered;
    title "Correlation Matrix for Filtered Dataset";
run;



/*Creating dataset by removing highly correlated vraibles*/
data WORK.CombinedData_2;
    set WORK.CombinedData_filtered (drop=MultipleLines_No OnlineSecurity_No OnlineBackup_No DeviceProtection_No TechSupport_N
run;
proc contents data=WORK.CombinedData_2;
run;
proc print data=WORK.CombinedData_2 (obs=5);
run;

/*Correlation of the filtered dataset*/
proc corr data=WORK.CombinedData_2 noprint outp=WORK.CorrelationMatrixFiltered;
run;

proc print data=WORK.CorrelationMatrixFiltered;
    title "Correlation Matrix for Filtered Dataset";
run;





/*Step 5: Explore The Data*/

/* Count of Churn Data Points */
proc sql;
    select count(*) as ChurnCount_1
    from WORK.CombinedData_2
    where Churn = 1; /* Assuming Churn is coded as 1 for churned */
    select count(*) as ChurnCount_0
    from WORK.CombinedData_2
    where Churn = 0; /* Assuming Churn is coded as 1 for churned */
quit;

/* Count and Percentage of Churned Customers */
proc freq data=WORK.CombinedData_2;
    tables Churn / out=ChurnTable noprint;
run;

proc sql;
    select Churn, count as ChurnCount, percent as ChurnPercent
    from ChurnTable;
quit;


/* Bar Graph of Churn (Yes vs. No) */
proc freq data=WORK.CombinedData_2 noprint;
    tables Churn / out=ChurnFreq;
run;

proc sgplot data=ChurnFreq;
    vbar Churn / response=Count stat=percent datalabel=percent datalabelpos=top;
    yaxis label="Percent";
    xaxis label="Churn" discreteorder=data;
    title "Churn Distribution";
run;

/* Contingency Table Between Churn and Gender */
proc freq data=WORK.CombinedData_2;
    tables gender_male*Churn / chisq norow nocol nopercent;
run;

/* Missing Value Counts for Each Variable */
proc means data=WORK.CombinedData_2 nmiss;
    output out=MissingCounts_filtered (drop=_type_ _freq_)
           nmiss=Num_Null_Values;
run;
proc print data=MissingCounts_filtered noobs;
run;

/* Stacked Bar of Tech Support and Churn */
proc freq data=WORK.CombinedData_2 noprint;
    tables TechSupport_Yes*Churn / out=TechSupport_ChurnFreq;
run;
proc freq data=WORK.CombinedData_2 noprint;
    tables TechSupport_Yes*Churn / out=TechSupport_ChurnFreq (rename=(count=Count));
run;
proc sgplot data=TechSupport_ChurnFreq;
    vbar TechSupport_Yes / response=Count group=Churn groupdisplay=stack;
    yaxis label="Count";
    xaxis label="Tech Support";
    keylegend / position=right;
run;

/* Stacked Bar of Senior Citizen and Churn */
proc freq data=WORK.CombinedData_2 noprint;
    tables SeniorCitizen*Churn / out=SeniorCitizen_ChurnFreq;
run;
proc freq data=WORK.CombinedData_2 noprint;
    tables SeniorCitizen*Churn / out=SeniorCitizen_Freq (rename=(count=Count));
run;
proc sgplot data=SeniorCitizen_ChurnFreq;
    vbar SeniorCitizen / response=Count group=Churn groupdisplay=stack;
    yaxis label="Count";
    xaxis label="SeniorCitizen";
    keylegend / position=right;
run;


/* Stacked Bar of Tenure and Churn */
proc freq data=WORK.CombinedData_2 noprint;
    tables Tenure*Churn / out=Tenure_ChurnFreq (rename=(count=ChurnCount));
run;
proc sgplot data=Tenure_ChurnFreq;
    vbar Tenure / response=ChurnCount group=Churn;
    yaxis label="Count";
    xaxis label="Tenure";
run;
```

```
/* Summary Statistics of Features */
proc means data=WORK.CombinedData_2 n mean std min max;
    var gender_male SeniorCitizen Partner Dependents Tenure PhoneService MultipleLines_Yes InternetService_FiberOptic Internet
run;

/* Histogram for Tenure */
proc sgplot data=WORK.CombinedData_2;
    histogram Tenure;
    density Tenure / type=kernel;
run;

/* Side-by-side Bar Charts for PaperlessBilling and PaymentMethods */
proc freq data=WORK.CombinedData_2 noprint;
    tables PaperlessBilling*Churn / out=PaperlessBilling_ChurnFreq (rename=(count=ChurnCount));
    tables PaymentMethod_CreditCard*Churn / out=CreditCard_ChurnFreq (rename=(count=ChurnCount));
    tables PaymentMethod_ElectronicCheck*Churn / out=ElectronicCheck_ChurnFreq (rename=(count=ChurnCount));
    tables PaymentMethod_MailedCheck*Churn / out=MailedCheck_ChurnFreq (rename=(count=ChurnCount));
run;

proc sgplot data=PaperlessBilling_ChurnFreq;
    vbar PaperlessBilling / response=ChurnCount group=Churn groupdisplay=cluster;
    yaxis label="Count";
    xaxis label="Paperless Billing";
run;

proc sgplot data=CreditCard_ChurnFreq;
    vbar PaymentMethod_CreditCard / response=ChurnCount group=Churn groupdisplay=cluster;
    yaxis label="Count";
    xaxis label="Payment Method- Credit Card";
run;

proc sgplot data=ElectronicCheck_ChurnFreq;
    vbar PaymentMethod_ElectronicCheck / response=ChurnCount group=Churn groupdisplay=cluster;
    yaxis label="Count";
    xaxis label="Payment Method - Electronic Check";
run;

proc sgplot data=MailedCheck_ChurnFreq;
    vbar PaymentMethod_MailedCheck / response=ChurnCount group=Churn groupdisplay=cluster;
    yaxis label="Count";
    xaxis label="Payment Method - Mailed Check";
run;


/* Side-by-side Bar Charts for PhoneService and InternetService */
proc freq data=WORK.CombinedData_2 noprint;
    tables PhoneService*Churn / out=PhoneService_ChurnFreq (rename=(count=ChurnCount));
    tables InternetService_FiberOptic*Churn / out=FiberOptic_ChurnFreq (rename=(count=ChurnCount));
    tables InternetService_No*Churn / out=NoInternet_ChurnFreq (rename=(count=ChurnCount));
run;

proc sgplot data=PhoneService_ChurnFreq;
    vbar PhoneService / response=ChurnCount group=Churn groupdisplay=cluster;
    yaxis label="Count";
    xaxis label="Phone Service";
run;

proc sgplot data=FiberOptic_ChurnFreq;
    vbar InternetService_FiberOptic / response=ChurnCount group=Churn groupdisplay=cluster;
    yaxis label="Count";
    xaxis label="Internet Service (Fiber Optic)";
run;

proc sgplot data=NoInternet_ChurnFreq;
    vbar InternetService_No / response=ChurnCount group=Churn groupdisplay=cluster;
    yaxis label="Count";
    xaxis label="Internet Service (No Internet)";
run;




/*Step 6: Model Building*/

/*Exclude `customerID` from the Analysis Dataset */
```

```
data WORK.CombinedData_Model;
    set WORK.CombinedData_2;
    drop customerID; /* Dropping customerID as it's not needed for the analysis */
run;

/*Sort the Data by `Churn` */
proc sort data=WORK.CombinedData_Model;
    by Churn;
run;

/*Split the Data into Training and Testing Sets */
proc surveyselect data=WORK.CombinedData_Model
    out=WORK.CombinedData_Split
    method=SRS /* Simple random sampling */
    samprate=0.7 /* 70% of data goes to the training set */
    outall
    seed=12345; /* Setting a seed for reproducibility */
    strata Churn; /* Ensuring stratification by Churn to maintain ratio */
run;

/* Create separate datasets for training and testing */
data WORK.Train WORK.Test;
    set WORK.CombinedData_Split;
    if Selected=1 then output WORK.Train;
    else output WORK.Test;
run;

/*Logistic Regression Modeling */
proc logistic data=WORK.Train descending outmodel=WORK.model;
    model Churn(event='1') = tenure MonthlyCharges SeniorCitizen Contract_OneYear Contract_TwoYear
                             PaymentMethod_CreditCard PaymentMethod_ElectronicCheck PaymentMethod_MailedCheck
                             InternetService_FiberOptic InternetService_No MultipleLines_Yes OnlineSecurity_Yes
                             OnlineBackup_Yes DeviceProtection_Yes TechSupport_Yes StreamingTV_Yes StreamingMovies_Yes
                             TotalCharges PhoneService PaperlessBilling Partner Dependents gender_male;
    output out=WORK.Logistic_Out p=preds; /* Output dataset with predicted probabilities */
run;

/*Scoring the Test Dataset Using the Model */
proc logistic inmodel=WORK.model;
    score data=WORK.Test out=WORK.Test_Scored;
run;

/* Categorize predicted probabilities to 0 or 1 based on threshold */
data WORK.Test_Scored;
    set WORK.Logistic_Out;
    if preds > 0.5 then pred_class = 1;
    else pred_class = 0;
run;

/* Creating the Confusion Matrix for Churn Predictions */
proc freq data=WORK.Test_Scored;
    tables Churn*pred_class / crosslist nocum nopercent;
    title "Confusion Matrix";
run;
title;

/*However There might be some over fitting going on in the model hence lets evaluate the VIFs*/
/* Check for multicollinearity by calculating VIF */
proc reg data=WORK.Train outvif;
    model Churn = tenure MonthlyCharges SeniorCitizen Contract_OneYear Contract_TwoYear
                  PaymentMethod_CreditCard PaymentMethod_ElectronicCheck PaymentMethod_MailedCheck
                  InternetService_FiberOptic InternetService_No MultipleLines_Yes OnlineSecurity_Yes
                  OnlineBackup_Yes DeviceProtection_Yes TechSupport_Yes StreamingTV_Yes StreamingMovies_Yes
                  TotalCharges PhoneService PaperlessBilling Partner Dependents gender_male / collinoint vif;
run;
quit;


/*Getting rid of variables with highly multicollinearity*/

data WORK.SelectedVariables;
    set WORK.CombinedData_2;
    keep tenure PaperlessBilling MonthlyCharges TotalCharges SeniorCitizen
        Contract_OneYear Contract_TwoYear PaymentMethod_CreditCard
        PaymentMethod_MailedCheck InternetService_FiberOptic InternetService_No
        MultipleLines_Yes TechSupport_Yes StreamingTV_Yes StreamingMovies_Yes Churn;
run;
```

```
/*Sort the Data by `Churn` */
proc sort data=WORK.SelectedVariables;
    by Churn;
run;

/*Split the Data into Training and Testing Sets */
proc surveyselect data=WORK.CombinedData_Model
    out=WORK.CombinedData_Split
    method=SRS /* Simple random sampling */
    samprate=0.7 /* 70% of data goes to the training set */
    outall
    seed=12345; /* Setting a seed for reproducibility */
    strata Churn; /* Ensuring stratification by Churn to maintain ratio */
run;

/* Create separate datasets for training and testing */
data WORK.Train WORK.Test;
    set WORK.CombinedData_Split;
    if Selected=1 then output WORK.Train;
    else output WORK.Test;
run;

/*Logistic Regression Modeling */
proc logistic data=WORK.Train descending outmodel=WORK.model;
    model Churn(event='1') = tenure PaperlessBilling MonthlyCharges TotalCharges SeniorCitizen
        Contract_OneYear Contract_TwoYear PaymentMethod_CreditCard
        PaymentMethod_MailedCheck InternetService_FiberOptic InternetService_No
        MultipleLines_Yes TechSupport_Yes StreamingTV_Yes StreamingMovies_Yes;
    output out=WORK.Logistic_Out p=preds; /* Output dataset with predicted probabilities */
run;

/*Scoring the Test Dataset Using the Model */
proc logistic inmodel=WORK.model;
    score data=WORK.Test out=WORK.Test_Scored;
run;

/* Categorize predicted probabilities to 0 or 1 based on threshold */
data WORK.Test_Scored;
    set WORK.Logistic_Out;
    if preds > 0.5 then pred_class = 1;
    else pred_class = 0;
run;

/* Creating the Confusion Matrix for Churn Predictions */
```