pt0xuaqcn

January 4, 2025

[ ]:
```
!pip install catboost

!pip uninstall -y scikit-learn
!pip install scikit-learn==1.3.1

!pip install imblearn
!pip install optuna
```

[2]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import OneHotEncoder
from sklearn.compose import ColumnTransformer
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import accuracy_score, confusion_matrix
from xgboost import XGBClassifier
from sklearn.svm import SVC
from catboost import CatBoostClassifier
from sklearn.model_selection import RandomizedSearchCV
from sklearn.model_selection import cross_val_score
from imblearn.over_sampling import SMOTE
from sklearn.metrics import classification_report
import optuna
```

[3]:
```
# Loading the dataset
file_path = './WA_Fn-UseC_-Telco-Customer-Churn.csv'

data = pd.read_csv(file_path)
print(f'dataset contains {data.shape[0]} rows and {data.shape[1]} columns')
```

```
dataset contains 7043 rows and 21 columns
```

[4]:
```
data.head(10)
```

[4]:
```
   customerID  gender  Senior_Citizen  Is_Married Dependents  tenure  \
0  7590-VHVEG  Female               0         Yes         No       1
1  5575-GNVDE    Male               0          No         No      34
2  3668-QPYBK    Male               0          No         No       2
3  7795-CFOCW    Male               0          No         No      45
4  9237-HQITU  Female               0          No         No       2
5  9305-CDSKC  Female               0          No         No       8
6  1452-KIOVK    Male               0          No        Yes      22
7  6713-OKOMC  Female               0          No         No      10
8  7892-POOKP  Female               0         Yes         No      28
9  6388-TABGU    Male               0          No        Yes      62

  Phone_Service              Dual Internet_Service Online_Security  …  \
0            No  No phone service              DSL              No  …
1           Yes                No              DSL             Yes  …
2           Yes                No              DSL             Yes  …
3            No  No phone service              DSL             Yes  …
4           Yes                No      Fiber optic              No  …
5           Yes               Yes      Fiber optic              No  …
6           Yes               Yes      Fiber optic              No  …
7            No  No phone service              DSL             Yes  …
8           Yes               Yes      Fiber optic              No  …
9           Yes                No              DSL             Yes  …

  Device_Protection Tech_Support Streaming_TV Streaming_Movies  \
0                No           No           No               No
1               Yes           No           No               No
2                No           No           No               No
3               Yes          Yes           No               No
4                No           No           No               No
5               Yes           No          Yes              Yes
6                No           No          Yes               No
7                No           No           No               No
8               Yes          Yes          Yes              Yes
9                No           No           No               No

         Contract Paperless_Billing             Payment_Method  \
0  Month-to-month               Yes           Electronic check
1        One year                No               Mailed check
2  Month-to-month               Yes               Mailed check
3        One year                No  Bank transfer (automatic)
4  Month-to-month               Yes           Electronic check
5  Month-to-month               Yes           Electronic check
6  Month-to-month               Yes    Credit card (automatic)
7  Month-to-month                No               Mailed check
8  Month-to-month               Yes           Electronic check
9        One year                No  Bank transfer (automatic)
```

```
  Monthly_Charges  Total_Charges Churn
0           29.85          29.85    No
1           56.95         1889.5    No
2           53.85         108.15   Yes
3           42.30        1840.75    No
4           70.70         151.65   Yes
5           99.65          820.5   Yes
6           89.10         1949.4    No
7           29.75          301.9    No
8          104.80        3046.05   Yes
9           56.15        3487.95    No

[10 rows x 21 columns]
```

[5]:
```
data.tail()
```

[5]:
```
      customerID  gender  Senior_Citizen  Is_Married Dependents  tenure  \
7038  6840-RESVB    Male               0         Yes        Yes      24
7039  2234-XADUH  Female               0         Yes        Yes      72
7040  4801-JZAZL  Female               0         Yes        Yes      11
7041  8361-LTMKD    Male               1         Yes         No       4
7042  3186-AJIEK    Male               0          No         No      66

     Phone_Service              Dual Internet_Service Online_Security  …  \
7038           Yes               Yes              DSL             Yes  …
7039           Yes               Yes      Fiber optic              No  …
7040            No  No phone service              DSL             Yes  …
7041           Yes               Yes      Fiber optic              No  …
7042           Yes                No      Fiber optic             Yes  …

     Device_Protection Tech_Support Streaming_TV Streaming_Movies  \
7038               Yes          Yes          Yes              Yes
7039               Yes           No          Yes              Yes
7040                No           No           No               No
7041                No           No           No               No
7042               Yes          Yes          Yes              Yes

            Contract Paperless_Billing             Payment_Method  \
7038        One year               Yes               Mailed check
7039        One year               Yes    Credit card (automatic)
7040  Month-to-month               Yes           Electronic check
7041  Month-to-month               Yes               Mailed check
7042        Two year               Yes  Bank transfer (automatic)

     Monthly_Charges  Total_Charges Churn
7038           84.80         1990.5    No
```

```
7039          103.20         7362.9    No
7040           29.60         346.45    No
7041           74.40          306.6   Yes
7042          105.65         6844.5    No

[5 rows x 21 columns]
```

[6]:
```python
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7043 entries, 0 to 7042
Data columns (total 21 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   customerID         7043 non-null   object
 1   gender             7043 non-null   object
 2   Senior_Citizen     7043 non-null   int64
 3   Is_Married         7043 non-null   object
 4   Dependents         7043 non-null   object
 5   tenure             7043 non-null   int64
 6   Phone_Service      7043 non-null   object
 7   Dual               7043 non-null   object
 8   Internet_Service   7043 non-null   object
 9   Online_Security    7043 non-null   object
 10  Online_Backup      7043 non-null   object
 11  Device_Protection  7043 non-null   object
 12  Tech_Support       7043 non-null   object
 13  Streaming_TV       7043 non-null   object
 14  Streaming_Movies   7043 non-null   object
 15  Contract           7043 non-null   object
 16  Paperless_Billing  7043 non-null   object
 17  Payment_Method     7043 non-null   object
 18  Monthly_Charges    7043 non-null   float64
 19  Total_Charges      7043 non-null   object
 20  Churn              7043 non-null   object
dtypes: float64(1), int64(2), object(18)
memory usage: 1.1+ MB
```

[7]:
```python
# Total_charges column should be of numerical type
data['Total_Charges'] = data['Total_Charges'].apply(pd.to_numeric,
 ↪errors='coerce') # invalid parsing will be set as NaN
```

[8]:
```python
data.isnull().sum()
```

[8]:
```
customerID           0
gender               0
Senior_Citizen       0
```

```
Is_Married           0
Dependents           0
tenure               0
Phone_Service        0
Dual                 0
Internet_Service     0
Online_Security      0
Online_Backup        0
Device_Protection    0
Tech_Support         0
Streaming_TV         0
Streaming_Movies     0
Contract             0
Paperless_Billing    0
Payment_Method       0
Monthly_Charges      0
Total_Charges       11
Churn                0
dtype: int64
```

[9]:
```
nan_rows = data[data['Total_Charges'].isna()]
print(nan_rows)
```

```
      customerID  gender  Senior_Citizen  Is_Married Dependents  tenure  \
488   4472-LVYGI  Female               0         Yes        Yes       0
753   3115-CZMZD    Male               0          No        Yes       0
936   5709-LVOEQ  Female               0         Yes        Yes       0
1082  4367-NUYAO    Male               0         Yes        Yes       0
1340  1371-DWPAZ  Female               0         Yes        Yes       0
3331  7644-OMVMY    Male               0         Yes        Yes       0
3826  3213-VVOLG    Male               0         Yes        Yes       0
4380  2520-SGTTA  Female               0         Yes        Yes       0
5218  2923-ARZLG    Male               0         Yes        Yes       0
6670  4075-WKNIU  Female               0         Yes        Yes       0
6754  2775-SEFEE    Male               0          No        Yes       0

     Phone_Service              Dual Internet_Service      Online_Security  \
488             No  No phone service              DSL                  Yes
753            Yes                No               No  No internet service
936            Yes                No              DSL                  Yes
1082           Yes               Yes               No  No internet service
1340            No  No phone service              DSL                  Yes
3331           Yes                No               No  No internet service
3826           Yes               Yes               No  No internet service
4380           Yes                No               No  No internet service
5218           Yes                No               No  No internet service
6670           Yes               Yes              DSL                   No
```

```
6754          Yes               Yes             DSL                    Yes

     …    Device_Protection         Tech_Support         Streaming_TV  \
488  …                  Yes                  Yes                  Yes
753  …  No internet service  No internet service  No internet service
936  …                  Yes                   No                  Yes
1082 …  No internet service  No internet service  No internet service
1340 …                  Yes                  Yes                  Yes
3331 …  No internet service  No internet service  No internet service
3826 …  No internet service  No internet service  No internet service
4380 …  No internet service  No internet service  No internet service
5218 …  No internet service  No internet service  No internet service
6670 …                  Yes                  Yes                  Yes
6754 …                   No                  Yes                   No

         Streaming_Movies  Contract Paperless_Billing  \
488                    No  Two year               Yes
753   No internet service  Two year                No
936                   Yes  Two year                No
1082  No internet service  Two year                No
1340                   No  Two year                No
3331  No internet service  Two year                No
3826  No internet service  Two year                No
4380  No internet service  Two year                No
5218  No internet service  One year               Yes
6670                   No  Two year                No
6754                   No  Two year               Yes

                 Payment_Method Monthly_Charges  Total_Charges  Churn
488   Bank transfer (automatic)           52.55            NaN     No
753                Mailed check           20.25            NaN     No
936                Mailed check           80.85            NaN     No
1082               Mailed check           25.75            NaN     No
1340    Credit card (automatic)           56.05            NaN     No
3331               Mailed check           19.85            NaN     No
3826               Mailed check           25.35            NaN     No
4380               Mailed check           20.00            NaN     No
5218               Mailed check           19.70            NaN     No
6670               Mailed check           73.35            NaN     No
6754  Bank transfer (automatic)           61.90            NaN     No

[11 rows x 21 columns]
```

[10]:
```
data = data.dropna(subset=['Total_Charges'])

data['Total_Charges'].isnull().sum()
```

[10]: 0

[11]:
```
data_dup = data.duplicated().any()
print(data_dup)
# data.drop_duplicates()
```

```
False
```

[12]:
```
data.describe()
```

[12]:

	Senior_Citizen	tenure	Monthly_Charges	Total_Charges
count	7032.000000	7032.000000	7032.000000	7032.000000
mean	0.162400	32.421786	64.798208	2283.300441
std	0.368844	24.545260	30.085974	2266.771362
min	0.000000	1.000000	18.250000	18.800000
25%	0.000000	9.000000	35.587500	401.450000
50%	0.000000	29.000000	70.350000	1397.475000
75%	0.000000	55.000000	89.862500	3794.737500
max	1.000000	72.000000	118.750000	8684.800000

[13]:
```
X = data.iloc[:, 1:-1] # Excluding the customerid and the churn
y = data.iloc[:, -1] # Churn

print(X.shape)
print(y.shape)
```

```
(7032, 19)
(7032,)
```

[14]:
```
X.head(5)
```

[14]:

	gender	Senior_Citizen	Is_Married	Dependents	tenure	Phone_Service	\
0	Female	0	Yes	No	1	No	
1	Male	0	No	No	34	Yes	
2	Male	0	No	No	2	Yes	
3	Male	0	No	No	45	No	
4	Female	0	No	No	2	Yes	

	Dual	Internet_Service	Online_Security	Online_Backup	\
0	No phone service	DSL	No	Yes	
1	No	DSL	Yes	No	
2	No	DSL	Yes	Yes	
3	No phone service	DSL	Yes	No	
4	No	Fiber optic	No	No	

	Device_Protection	Tech_Support	Streaming_TV	Streaming_Movies	\
0	No	No	No	No	

```
1                Yes          No          No                No
2                 No          No          No                No
3                Yes         Yes          No                No
4                 No          No          No                No

         Contract Paperless_Billing             Payment_Method  \
0  Month-to-month               Yes           Electronic check
1        One year                No               Mailed check
2  Month-to-month               Yes               Mailed check
3        One year                No  Bank transfer (automatic)
4  Month-to-month               Yes           Electronic check

   Monthly_Charges  Total_Charges
0            29.85          29.85
1            56.95        1889.50
2            53.85         108.15
3            42.30        1840.75
4            70.70         151.65
```

[15]:

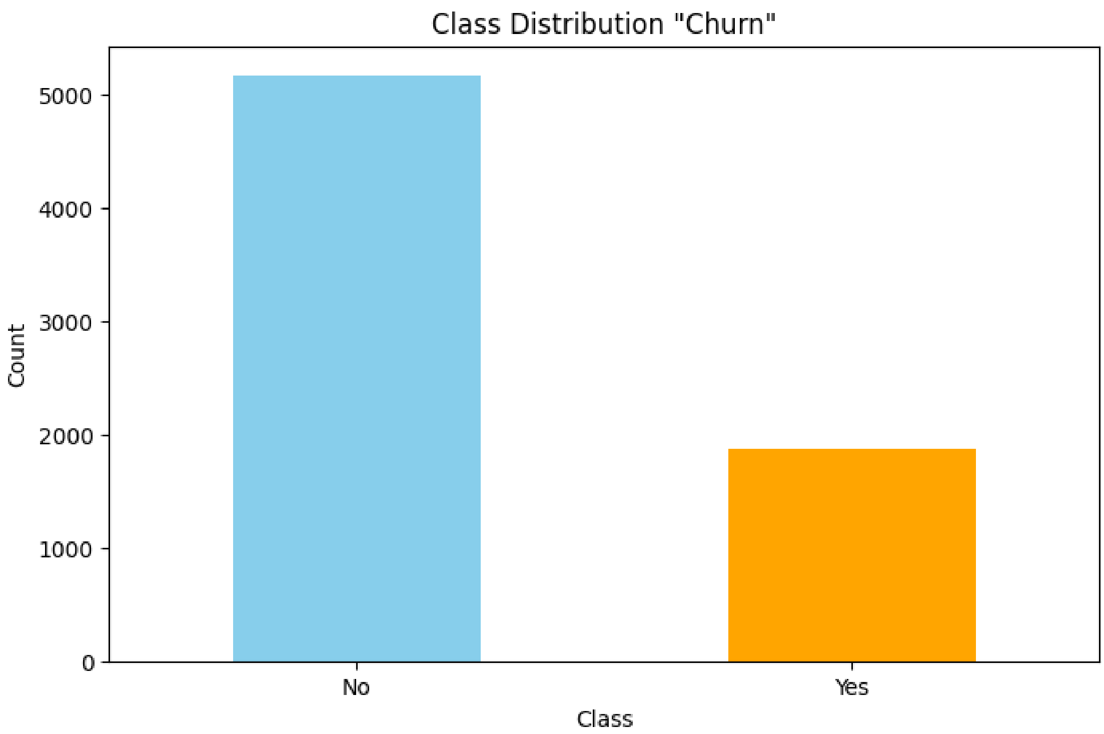
```python
y.head(5)
```

[15]:

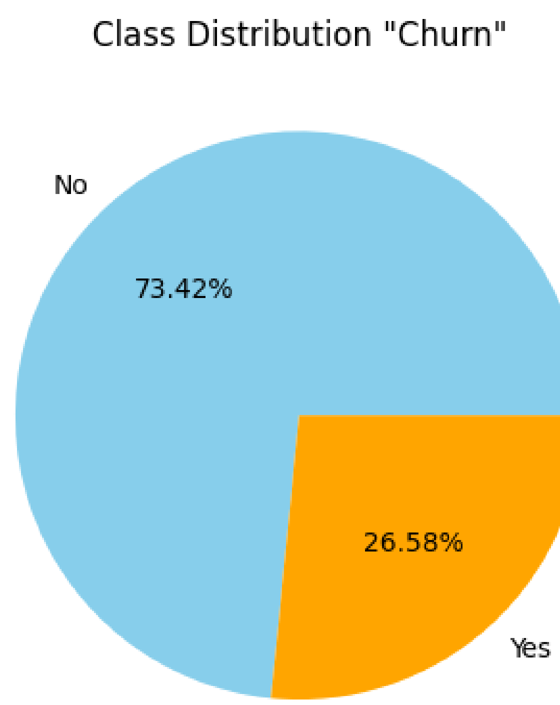
```
0     No
1     No
2    Yes
3     No
4    Yes
Name: Churn, dtype: object
```

[16]:

```python
# Get the count of each class
class_counts = y.value_counts() # Counts of unique values

# Plot the counts as a bar chart
plt.figure(figsize=(8, 5))
class_counts.plot(kind='bar', color=['skyblue', 'orange'])
plt.title('Class Distribution "Churn"')
plt.xlabel('Class')
plt.ylabel('Count')
plt.xticks(rotation=0)
plt.show()
```



[17]:
```
class_counts.plot.pie(autopct='%1.2f%%', colors=['skyblue', 'orange'])
plt.title('Class Distribution "Churn"')
plt.ylabel('')
plt.show()
# Looks Like an Imbalanced Dataset
```



[18]:
```
def check_unique():
    # Checking unique values to choose which technique to apply
    should_be_one_hot_encoded = []
    should_be_label_encoded = []

    for col in X.columns:
        if X[col].dtypes == 'object': # Exclude numerical values
            print(f'{col}: {X[col].unique()}')
            if len(X[col].unique()) > 2:
                should_be_one_hot_encoded.append(col)
            else:
                should_be_label_encoded.append(col)

    print('\nOne-Hot Encoded : ', should_be_one_hot_encoded, '\n')
    print('Label Encoded : ', should_be_label_encoded)
    return should_be_one_hot_encoded, should_be_label_encoded

should_be_one_hot_encoded, should_be_label_encoded = check_unique()
```

```
gender: ['Female' 'Male']
Is_Married: ['Yes' 'No']
Dependents: ['No' 'Yes']
```

```
Phone_Service: ['No' 'Yes']
Dual: ['No phone service' 'No' 'Yes']
Internet_Service: ['DSL' 'Fiber optic' 'No']
Online_Security: ['No' 'Yes' 'No internet service']
Online_Backup: ['Yes' 'No' 'No internet service']
Device_Protection: ['No' 'Yes' 'No internet service']
Tech_Support: ['No' 'Yes' 'No internet service']
Streaming_TV: ['No' 'Yes' 'No internet service']
Streaming_Movies: ['No' 'Yes' 'No internet service']
Contract: ['Month-to-month' 'One year' 'Two year']
Paperless_Billing: ['Yes' 'No']
Payment_Method: ['Electronic check' 'Mailed check' 'Bank transfer (automatic)'
 'Credit card (automatic)']

One-Hot Encoded :  ['Dual', 'Internet_Service', 'Online_Security',
'Online_Backup', 'Device_Protection', 'Tech_Support', 'Streaming_TV',
'Streaming_Movies', 'Contract', 'Payment_Method']

Label Encoded :  ['gender', 'Is_Married', 'Dependents', 'Phone_Service',
'Paperless_Billing']
```

[19]:

```
# Hidden Redundancy
# Columns that has the 'No Internet Service'
cols = [col for col in X.columns if 'No internet service' in X[col].unique()]
print(cols)

for col in cols:
    X[col] = X[col].replace('No internet service', 'No')
    X[col] = X[col].replace('No phone service', 'No')
```

```
['Online_Security', 'Online_Backup', 'Device_Protection', 'Tech_Support',
'Streaming_TV', 'Streaming_Movies']
```

[20]:

```
check_unique()
```

```
gender: ['Female' 'Male']
Is_Married: ['Yes' 'No']
Dependents: ['No' 'Yes']
Phone_Service: ['No' 'Yes']
Dual: ['No phone service' 'No' 'Yes']
Internet_Service: ['DSL' 'Fiber optic' 'No']
Online_Security: ['No' 'Yes']
Online_Backup: ['Yes' 'No']
Device_Protection: ['No' 'Yes']
Tech_Support: ['No' 'Yes']
Streaming_TV: ['No' 'Yes']
Streaming_Movies: ['No' 'Yes']
Contract: ['Month-to-month' 'One year' 'Two year']
```

```
Paperless_Billing: ['Yes' 'No']
Payment_Method: ['Electronic check' 'Mailed check' 'Bank transfer (automatic)'
 'Credit card (automatic)']

One-Hot Encoded :  ['Dual', 'Internet_Service', 'Contract', 'Payment_Method']

Label Encoded :  ['gender', 'Is_Married', 'Dependents', 'Phone_Service',
'Online_Security', 'Online_Backup', 'Device_Protection', 'Tech_Support',
'Streaming_TV', 'Streaming_Movies', 'Paperless_Billing']
```

[20]:
```
(['Dual', 'Internet_Service', 'Contract', 'Payment_Method'],
 ['gender',
  'Is_Married',
  'Dependents',
  'Phone_Service',
  'Online_Security',
  'Online_Backup',
  'Device_Protection',
  'Tech_Support',
  'Streaming_TV',
  'Streaming_Movies',
  'Paperless_Billing'])
```

[21]:
```
le = LabelEncoder()
for col in should_be_label_encoded:
    X[col] = le.fit_transform(X[col]) # Apply label encoding for each column

for col in should_be_label_encoded:
    print(f'{col}: {X[col].unique()}')

# Label Encoding the Target
y = le.fit_transform(y)
print(y)
```

```
gender: [0 1]
Is_Married: [1 0]
Dependents: [0 1]
Phone_Service: [0 1]
Paperless_Billing: [1 0]
[0 0 1 … 0 1 0]
```

[22]:
```
# Get the indexes of columns to transform
hot_encode_indexes = X.columns.get_indexer(should_be_one_hot_encoded)
print(hot_encode_indexes)
ct = ColumnTransformer(transformers=[('encoder', OneHotEncoder(),␣
 ↪hot_encode_indexes)], remainder='passthrough')
# Fit and transform the data
```

```
X_transformed = np.array(ct.fit_transform(X))
print(X_transformed)
```

```
[ 6  7  8  9 10 11 12 13 14 16]
[[0.0000e+00 1.0000e+00 0.0000e+00 … 1.0000e+00 2.9850e+01 2.9850e+01]
 [1.0000e+00 0.0000e+00 0.0000e+00 … 0.0000e+00 5.6950e+01 1.8895e+03]
 [1.0000e+00 0.0000e+00 0.0000e+00 … 1.0000e+00 5.3850e+01 1.0815e+02]
 …
 [0.0000e+00 1.0000e+00 0.0000e+00 … 1.0000e+00 2.9600e+01 3.4645e+02]
 [0.0000e+00 0.0000e+00 1.0000e+00 … 1.0000e+00 7.4400e+01 3.0660e+02]
 [1.0000e+00 0.0000e+00 0.0000e+00 … 1.0000e+00 1.0565e+02 6.8445e+03]]
```

[23]:
```
# Get the feature names for the one-hot encoded columns
encoder = ct.transformers_[0][1]  # The encoder used for one-hot encoding
encoded_feature_names = encoder.
 ↪get_feature_names_out(input_features=should_be_one_hot_encoded)

# Create a DataFrame with the transformed data
# Concatenate the new one-hot encoded feature names and original columns that␣
 ↪weren't transformed
X_transformed_df = pd.DataFrame(X_transformed, columns=np.
 ↪concatenate([encoded_feature_names, X.columns.
 ↪difference(should_be_one_hot_encoded)]))

# Show the resulting DataFrame
print(X_transformed_df)
```

```
      Dual_No  Dual_No phone service  Dual_Yes  Internet_Service_DSL  \
0         0.0                    1.0       0.0                   1.0
1         1.0                    0.0       0.0                   1.0
2         1.0                    0.0       0.0                   1.0
3         0.0                    1.0       0.0                   1.0
4         1.0                    0.0       0.0                   0.0
…         …                     …        …                    …
7027      0.0                    0.0       1.0                   1.0
7028      0.0                    0.0       1.0                   0.0
7029      0.0                    1.0       0.0                   1.0
7030      0.0                    0.0       1.0                   0.0
7031      1.0                    0.0       0.0                   0.0

      Internet_Service_Fiber optic  Internet_Service_No  Online_Security_No  \
0                              0.0                  0.0                 1.0
1                              0.0                  0.0                 0.0
2                              0.0                  0.0                 0.0
3                              0.0                  0.0                 0.0
4                              1.0                  0.0                 1.0
…                             …                    …                  …
```

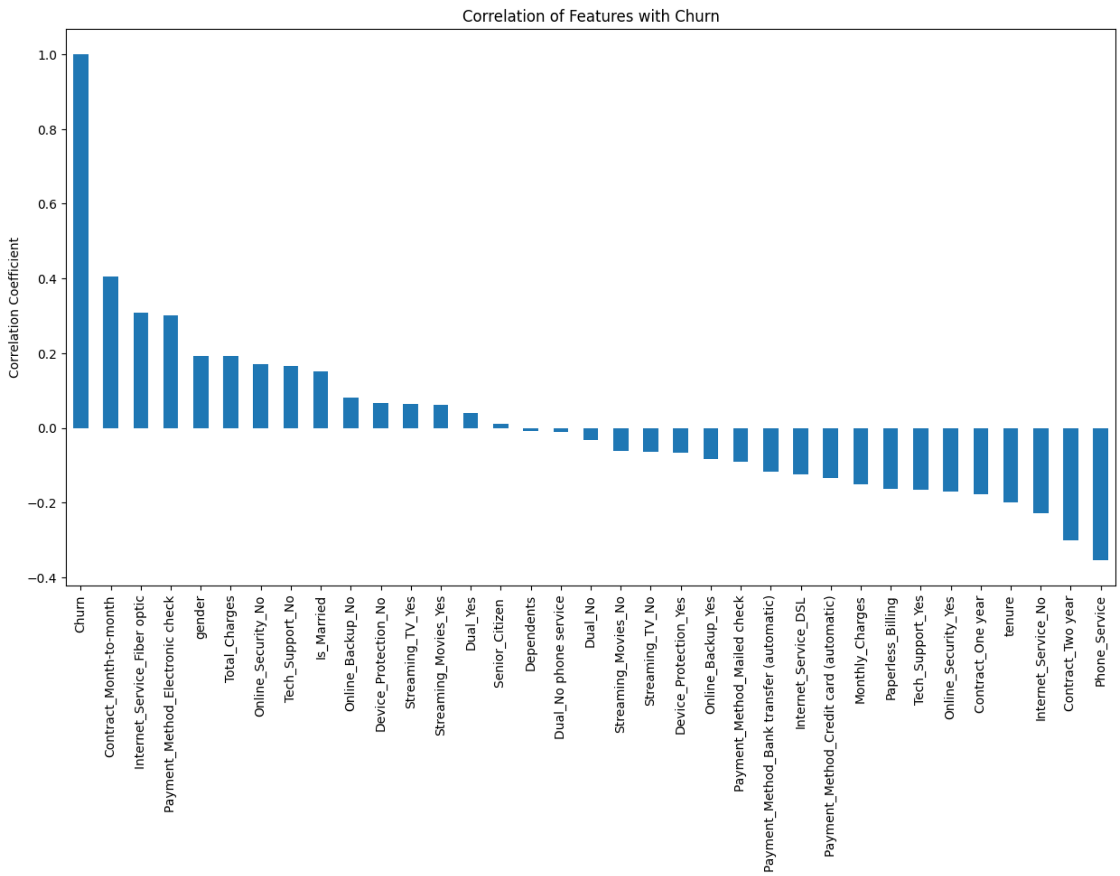
```
7027                          0.0                 0.0                0.0
7028                          1.0                 0.0                1.0
7029                          0.0                 0.0                0.0
7030                          1.0                 0.0                1.0
7031                          1.0                 0.0                0.0

      Online_Security_Yes  Online_Backup_No  Online_Backup_Yes  …  \
0                     0.0               0.0                1.0  …
1                     1.0               1.0                0.0  …
2                     1.0               0.0                1.0  …
3                     1.0               1.0                0.0  …
4                     0.0               1.0                0.0  …
…                     …                 …                  …    …
7027                  1.0               1.0                0.0  …
7028                  0.0               0.0                1.0  …
7029                  1.0               1.0                0.0  …
7030                  0.0               1.0                0.0  …
7031                  1.0               1.0                0.0  …

      Payment_Method_Mailed check  Dependents  Is_Married  Monthly_Charges  \
0                             0.0         0.0         0.0              1.0
1                             1.0         1.0         0.0              0.0
2                             1.0         1.0         0.0              0.0
3                             0.0         1.0         0.0              0.0
4                             0.0         0.0         0.0              0.0
…                             …           …           …                …
7027                          1.0         1.0         0.0              1.0
7028                          0.0         0.0         0.0              1.0
7029                          0.0         0.0         0.0              1.0
7030                          1.0         1.0         1.0              1.0
7031                          0.0         1.0         0.0              0.0

      Paperless_Billing  Phone_Service  Senior_Citizen  Total_Charges  \
0                   0.0            1.0             0.0            1.0
1                   0.0           34.0             1.0            0.0
2                   0.0            2.0             1.0            1.0
3                   0.0           45.0             0.0            0.0
4                   0.0            2.0             1.0            1.0
…                   …              …               …              …
7027                1.0           24.0             1.0            1.0
7028                1.0           72.0             1.0            1.0
7029                1.0           11.0             0.0            1.0
7030                0.0            4.0             1.0            1.0
7031                0.0           66.0             1.0            1.0

      gender   tenure
0      29.85    29.85
1      56.95  1889.50
```

```
2       53.85   108.15
3       42.30  1840.75
4       70.70   151.65
…       …       …
7027    84.80  1990.50
7028   103.20  7362.90
7029    29.60   346.45
7030    74.40   306.60
7031   105.65  6844.50

[7032 rows x 34 columns]
```

[24]:

```python
#Get Correlation of "Churn" with other variables:
plt.figure(figsize=(15,8))
y_df = pd.DataFrame(y, columns=['Churn'])

# Concatenate the feature DataFrame and the target DataFrame
new_df = pd.concat([X_transformed_df, y_df], axis=1)

# Calculate correlations with the 'Churn' column
correlation = new_df.corr()['Churn'].sort_values(ascending=False)

# Plot the correlation of Churn with other variables
plt.figure(figsize=(15, 8))
correlation.plot(kind='bar')
plt.title("Correlation of Features with Churn")
plt.ylabel("Correlation Coefficient")
plt.show()
```

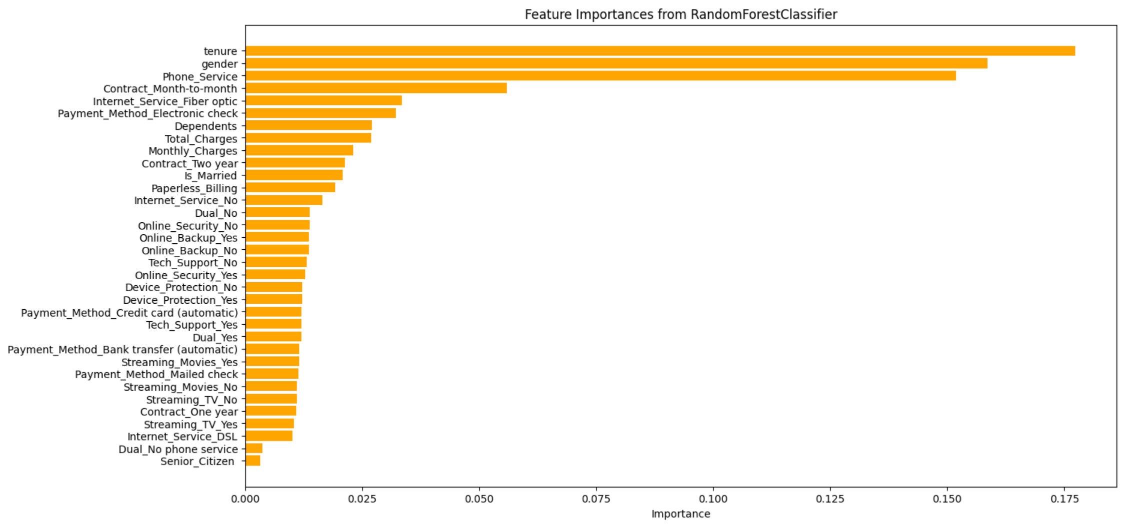
```
<Figure size 1500x800 with 0 Axes>
```



[25]:
```python
# Dataset has mixed values between numerical and categorical so its best to use␣
↪randomforest rather than correlation matrix or chi-square
model = RandomForestClassifier()
model.fit(X_transformed, y)

# Get feature importances
importances = model.feature_importances_
print(importances)

# Create a DataFrame to store feature names and their importances
features_df = pd.DataFrame({
    'Feature': X_transformed_df.columns,
    'Importance': importances
})

# Sort the features by importance in descending order
features_df = features_df.sort_values(by='Importance', ascending=True)
```

```python
# Plot the feature importances
plt.figure(figsize=(15, 8))
plt.barh(features_df['Feature'], features_df['Importance'], color='orange')
plt.xlabel('Importance')
plt.title('Feature Importances from RandomForestClassifier')
plt.show()
```

```
[0.01374715 0.00369137 0.01199075 0.01005996 0.03341913 0.01647469
 0.01370785 0.0127577  0.01357925 0.013618   0.01220072 0.01215452
 0.01309365 0.01201278 0.01098046 0.01042185 0.0110465  0.01150989
 0.05592667 0.01096662 0.0213097  0.0115707  0.01205327 0.03225918
 0.0113292  0.02705985 0.02080155 0.02300854 0.01930369 0.15190371
 0.00313831 0.02691079 0.15863367 0.17735834]
```



[26]:
```python
number_of_features = 16 # Take highest 15 features
filtered_features = []
for feature in features_df.tail(number_of_features).Feature:
    print(feature)
    filtered_features.append(feature)

for feature in X_transformed_df.columns:
    if feature not in filtered_features:
        X_transformed_df = X_transformed_df.drop(feature, axis=1)

print(X_transformed_df)
```

```
Online_Backup_Yes
Online_Security_No
Dual_No
Internet_Service_No
```

```
Paperless_Billing
Is_Married
Contract_Two year
Monthly_Charges
Total_Charges
Dependents
Payment_Method_Electronic check
Internet_Service_Fiber optic
Contract_Month-to-month
Phone_Service
gender
tenure
      Dual_No  Internet_Service_Fiber optic  Internet_Service_No  \
0         0.0                           0.0                  0.0
1         1.0                           0.0                  0.0
2         1.0                           0.0                  0.0
3         0.0                           0.0                  0.0
4         1.0                           1.0                  0.0
…         …                            …                   …
7027      0.0                           0.0                  0.0
7028      0.0                           1.0                  0.0
7029      0.0                           0.0                  0.0
7030      0.0                           1.0                  0.0
7031      1.0                           1.0                  0.0

      Online_Security_No  Online_Backup_Yes  Contract_Month-to-month  \
0                    1.0                1.0                      1.0
1                    0.0                0.0                      0.0
2                    0.0                1.0                      1.0
3                    0.0                0.0                      0.0
4                    1.0                0.0                      1.0
…                   …                  …                       …
7027                 0.0                0.0                      0.0
7028                 1.0                1.0                      0.0
7029                 0.0                0.0                      1.0
7030                 1.0                0.0                      1.0
7031                 0.0                0.0                      0.0

      Contract_Two year  Payment_Method_Electronic check  Dependents  \
0                   0.0                              1.0         0.0
1                   0.0                              0.0         1.0
2                   0.0                              0.0         1.0
3                   0.0                              0.0         1.0
4                   0.0                              1.0         0.0
…                  …                              …          …
7027                0.0                              0.0         1.0
7028                0.0                              0.0         0.0
7029                0.0                              1.0         0.0
```

```
7030              0.0                               0.0         1.0
7031              1.0                               0.0         1.0

      Is_Married  Monthly_Charges  Paperless_Billing  Phone_Service  \
0            0.0              1.0                0.0            1.0
1            0.0              0.0                0.0           34.0
2            0.0              0.0                0.0            2.0
3            0.0              0.0                0.0           45.0
4            0.0              0.0                0.0            2.0
…            …                …                  …              …
7027         0.0              1.0                1.0           24.0
7028         0.0              1.0                1.0           72.0
7029         0.0              1.0                1.0           11.0
7030         1.0              1.0                0.0            4.0
7031         0.0              0.0                0.0           66.0

      Total_Charges  gender   tenure
0               1.0   29.85    29.85
1               0.0   56.95  1889.50
2               1.0   53.85   108.15
3               0.0   42.30  1840.75
4               1.0   70.70   151.65
…               …       …       …
7027            1.0   84.80  1990.50
7028            1.0  103.20  7362.90
7029            1.0   29.60   346.45
7030            1.0   74.40   306.60
7031            1.0  105.65  6844.50

[7032 rows x 16 columns]
```

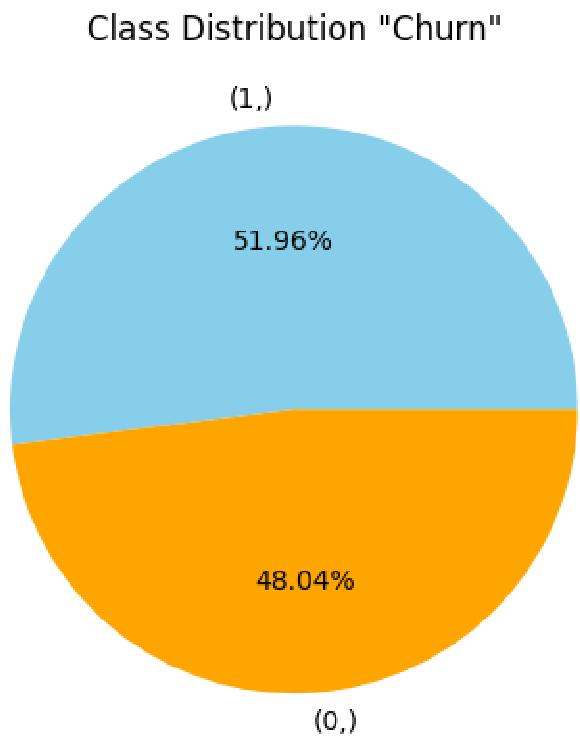
[27]:
```python
X_train, X_test, y_train, y_test = train_test_split(np.array(X_transformed_df),␣
    ↪y, test_size=0.3, random_state=42)
```

[28]:
```python
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)
```

[29]:
```python
from imblearn.combine import SMOTEENN
# from imblearn.over_sampling import SMOTE
smote = SMOTEENN(sampling_strategy='auto', random_state=42)
X_train, y_train = smote.fit_resample(X_train, y_train)
```

[30]:
```python
pd.DataFrame(y_train).value_counts().plot.pie(autopct='%1.2f%%',␣
    ↪colors=['skyblue', 'orange'])
plt.title('Class Distribution "Churn"')
plt.ylabel('')
```

```
plt.show()
```



[31]:
```
# computational intensive with catboost
# # from sklearn.metrics import accuracy_score
# def objective(trial):
#     params = {
#         'n_estimators': trial.suggest_int('n_estimators', 50, 200),
#         'learning_rate': trial.suggest_loguniform('learning_rate', 0.01, 0.3),
#         'max_depth': trial.suggest_int('max_depth', 3, 9),
#     }
#     model = CatBoostClassifier(**params)
#     # model.fit(X_train, y_train)
#     # Evaluate on the test set
#     # test_accuracy = accuracy_score(y_test, model.predict(X_test))
#     # return test_accuracy
#     return cross_val_score(model, X_train, y_train, cv=5, scoring='accuracy').
↪mean()
# study = optuna.create_study(direction='maximize')
# study.optimize(objective, n_trials=50)
# print(study.best_params)
```

[32]:
```
# classifier = XGBClassifier(n_estimators=174, learning_rate=0.
 ↪24764189932721647, max_depth=9)
# classifier = LGBMClassifier(learning_rate= 0.1841164348061631, max_depth = 9,␣
 ↪n_estimators = 199, verbose = -1)
# from lightgbm import LGBMClassifier
from sklearn.linear_model import LogisticRegression
classifier = CatBoostClassifier(silent=True, random_state=2)
classifier.fit(X_train, y_train)
```

[32]: <catboost.core.CatBoostClassifier at 0x7fd376cafd60>

[33]:
```
from sklearn.metrics import confusion_matrix, accuracy_score
y_pred = classifier.predict(X_test)
cm = confusion_matrix(y_test, y_pred)
print(cm)
print(accuracy_score(y_test, y_pred))
```

```
[[1151  398]
 [ 137  424]]
0.7464454976303317
```

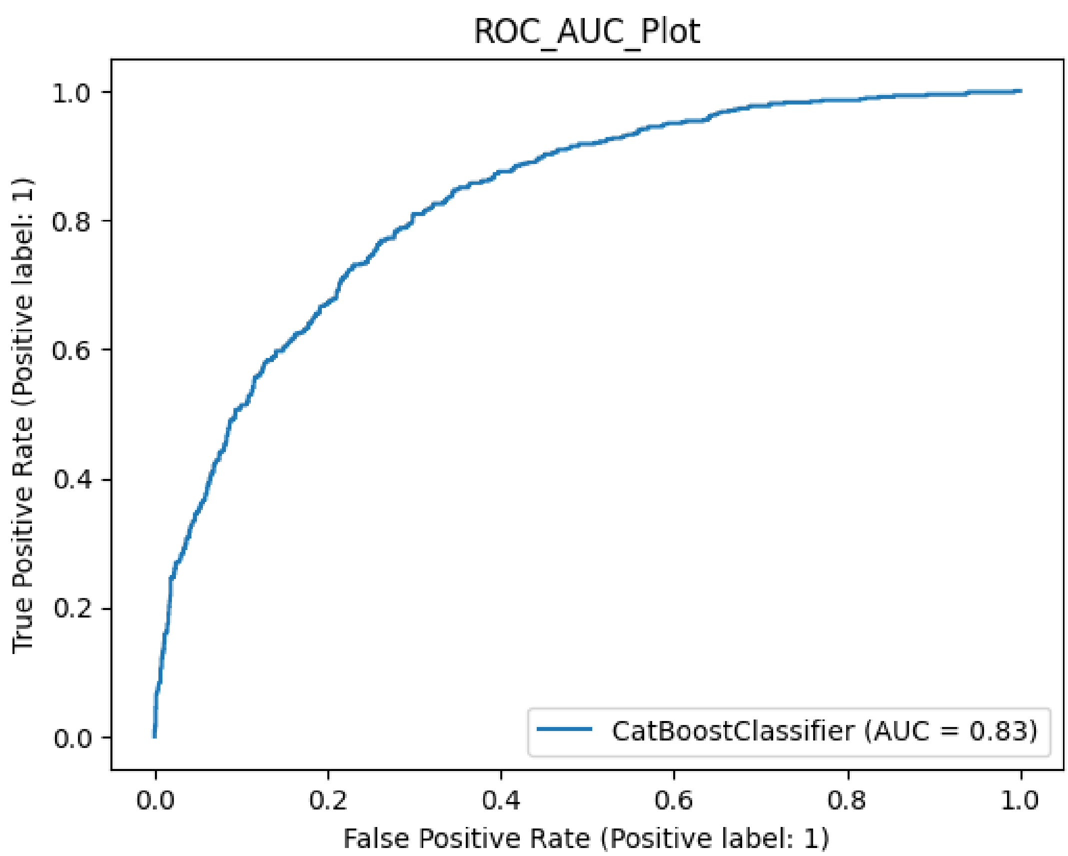
[34]:
```
print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

           0       0.89      0.74      0.81      1549
           1       0.52      0.76      0.61       561

    accuracy                           0.75      2110
   macro avg       0.70      0.75      0.71      2110
weighted avg       0.79      0.75      0.76      2110
```

[35]:
```
from sklearn.metrics import roc_auc_score, RocCurveDisplay
from sklearn.model_selection import RepeatedStratifiedKFold, cross_val_score

def model_evaluation_roc(classifier, x_train, y_train, x_test, y_test):
    # Fit the classifier
    classifier.fit(x_train, y_train)

    # Predict on the test set
    prediction = classifier.predict(x_test)

    # Cross-validation score
    cv = RepeatedStratifiedKFold(n_splits=10, n_repeats=3, random_state=1)
    cv_score = cross_val_score(classifier, x_train, y_train, cv=cv,␣
 ↪scoring='roc_auc').mean()
```

```
    print("Cross Validation Score : ", '{0:.2%}'.format(cv_score))

    # ROC AUC score
    roc_auc = roc_auc_score(y_test, prediction)
    print("ROC_AUC Score : ", '{0:.2%}'.format(roc_auc))

    # Plot ROC Curve
    RocCurveDisplay.from_estimator(classifier, x_test, y_test)
    plt.title('ROC_AUC_Plot')
    plt.show()

# Call the function
# Replace `classifier` with an instance of your model, e.g.,␣
↪LogisticRegression(), RandomForestClassifier(), etc.
# X_train, y_train, X_test, y_test should be your training and testing datasets
model_evaluation_roc(classifier, X_train, y_train, X_test, y_test)
```

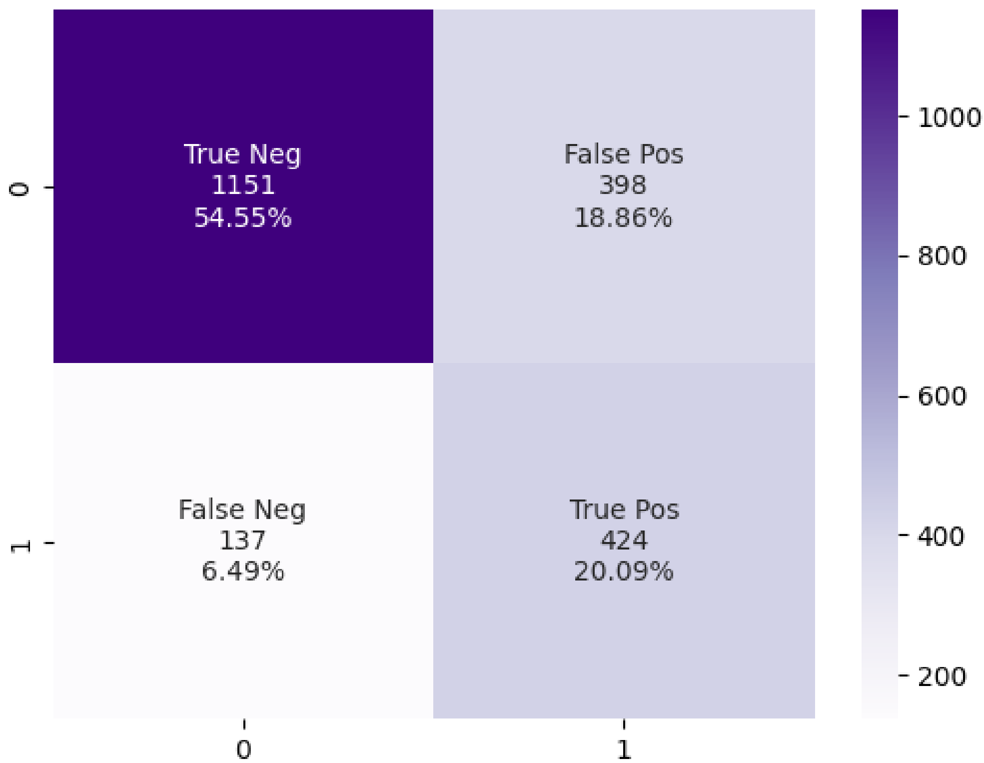
```
Cross Validation Score :  99.40%
ROC_AUC Score :  74.94%
```



[36]:

```
def model_evaluation_cm(classifier,x_test,y_test):

    # Confusion Matrix
    cm = confusion_matrix(y_test,classifier.predict(x_test))
    names = ['True Neg','False Pos','False Neg','True Pos']
    counts = [value for value in cm.flatten()]
    percentages = ['{0:.2%}'.format(value) for value in cm.flatten()/np.sum(cm)]
    labels = [f'{v1}\n{v2}\n{v3}' for v1, v2, v3 in␣
 ↪zip(names,counts,percentages)]
    labels = np.asarray(labels).reshape(2,2)
    sns.heatmap(cm,annot = labels,cmap = 'Purples',fmt ='')

    # Classification Report
    print(classification_report(y_test,classifier.predict(x_test)))
model_evaluation_cm(classifier, X_test, y_test)
```

```
              precision    recall  f1-score   support

           0       0.89      0.74      0.81      1549
           1       0.52      0.76      0.61       561

    accuracy                           0.75      2110
   macro avg       0.70      0.75      0.71      2110
weighted avg       0.79      0.75      0.76      2110
```



[37]:
```python
import pickle

# save
with open('catboost_model.pkl','wb') as f:
    pickle.dump(classifier,f)
```