In [1]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as pltj
import matplotlib.ticker as mtick
```

In [2]:
```
telecom_cust=pd.read_csv
("C:\\Users\\SRI KAAVYA\\Downloads\\churn pred in telecom.zip")
```

In [3]:
```
telecom_cust.head(19)
```

Out[3]:

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines
0	7590-VHVEG	Female	0	Yes	No	1	No	No phone service
1	5575-GNVDE	Male	0	No	No	34	Yes	No
2	3668-QPYBK	Male	0	No	No	2	Yes	No
3	7795-CFOCW	Male	0	No	No	45	No	No phone service
4	9237-HQITU	Female	0	No	No	2	Yes	No
5	9305-CDSKC	Female	0	No	No	8	Yes	Yes
6	1452-KIOVK	Male	0	No	Yes	22	Yes	Yes
7	6713-OKOMC	Female	0	No	No	10	No	No phone service
8	7892-POOKP	Female	0	Yes	No	28	Yes	Yes
9	6388-TABGU	Male	0	No	Yes	62	Yes	No
10	9763-GRSKD	Male	0	Yes	Yes	13	Yes	No
11	7469-LKBCI	Male	0	No	No	16	Yes	No
12	8091-TTVAX	Male	0	Yes	No	58	Yes	Yes
13	0280-XJGEX	Male	0	No	No	49	Yes	Yes
14	5129-JLPIS	Male	0	No	No	25	Yes	No
15	3655-SNQYZ	Female	0	Yes	Yes	69	Yes	Yes
16	8191-XWSZG	Female	0	No	No	52	Yes	No
17	9959-WOFKT	Male	0	No	Yes	71	Yes	Yes
18	4190-MFLUW	Female	0	Yes	Yes	10	Yes	No

19 rows × 21 columns

In [4]:
```
telecom_cust.columns.values
```

Out[4]:
```
array(['customerID', 'gender', 'SeniorCitizen', 'Partner', 'Dependents',
       'tenure', 'PhoneService', 'MultipleLines', 'InternetService',
       'OnlineSecurity', 'OnlineBackup', 'DeviceProtection',
       'TechSupport', 'StreamingTV', 'StreamingMovies', 'Contract',
       'PaperlessBilling', 'PaymentMethod', 'MonthlyCharges',
       'TotalCharges', 'Churn'], dtype=object)
```

In [5]:
```
telecom_cust.dtypes
```

Out[5]:
```
customerID           object
gender               object
SeniorCitizen         int64
Partner              object
Dependents           object
tenure                int64
PhoneService         object
MultipleLines        object
InternetService      object
OnlineSecurity       object
OnlineBackup         object
DeviceProtection     object
TechSupport          object
StreamingTV          object
StreamingMovies      object
Contract             object
PaperlessBilling     object
PaymentMethod        object
MonthlyCharges      float64
TotalCharges         object
Churn                object
dtype: object
```

In [6]:
```
telecom_cust.TotalCharges=pd.to_numeric(telecom_cust.TotalCharges,errors='coer
telecom_cust.isnull().sum()
```

Out[6]:
```
customerID           0
gender               0
SeniorCitizen        0
Partner              0
Dependents           0
tenure               0
PhoneService         0
MultipleLines        0
InternetService      0
OnlineSecurity       0
OnlineBackup         0
DeviceProtection     0
TechSupport          0
StreamingTV          0
StreamingMovies      0
Contract             0
PaperlessBilling     0
PaymentMethod        0
MonthlyCharges       0
TotalCharges        11
Churn                0
dtype: int64
```

In [7]:
```
telecom_cust.dropna(inplace = True)
df2 = telecom_cust.iloc[:,1:]
df2['Churn'].replace(to_replace='Yes', value=1, inplace=True)
df2['Churn'].replace(to_replace='No',  value=0, inplace=True)
df_dummies = pd.get_dummies(df2)
df_dummies.head()
```

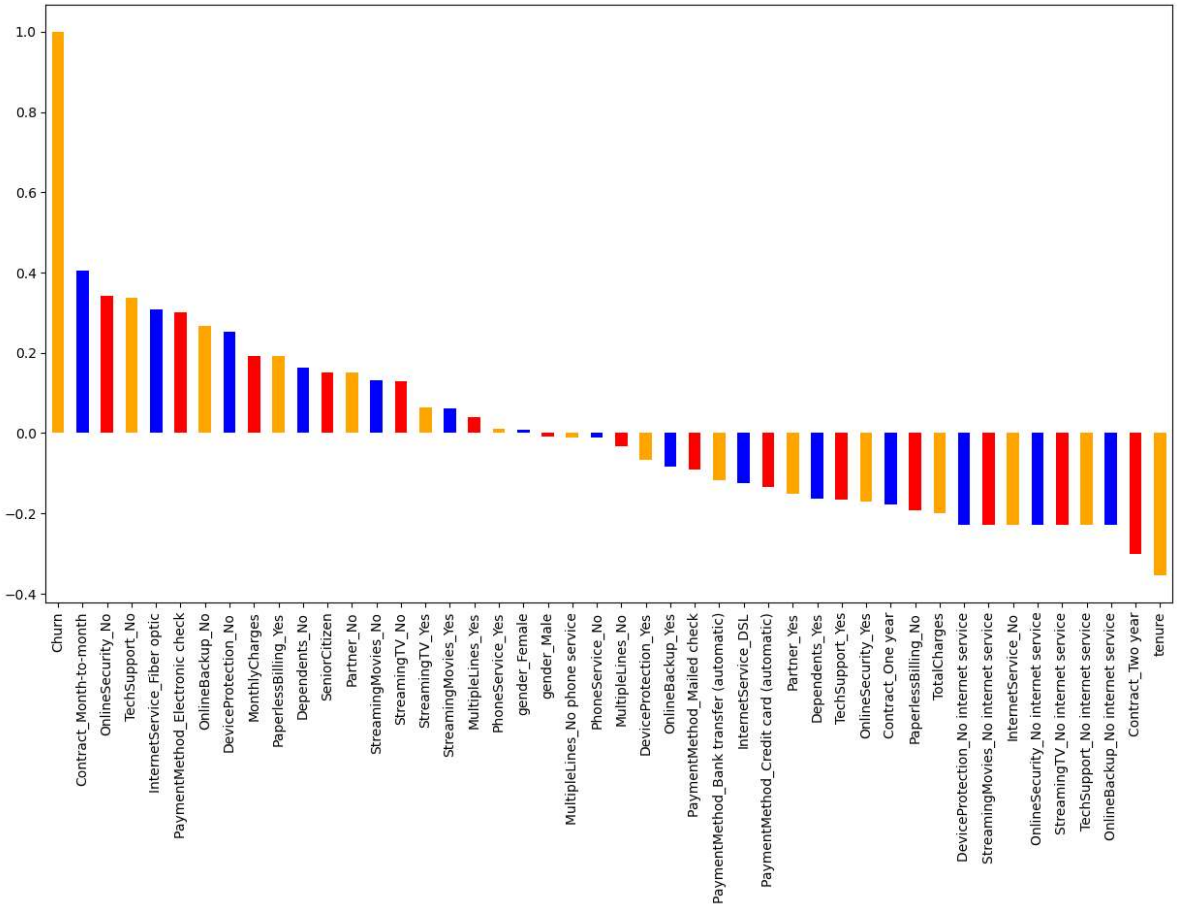
Out[7]:

	SeniorCitizen	tenure	MonthlyCharges	TotalCharges	Churn	gender_Female	gender_Male	P
0	0	1	29.85	29.85	0	1	0	
1	0	34	56.95	1889.50	0	0	1	
2	0	2	53.85	108.15	1	0	1	
3	0	45	42.30	1840.75	0	0	1	
4	0	2	70.70	151.65	1	1	0	

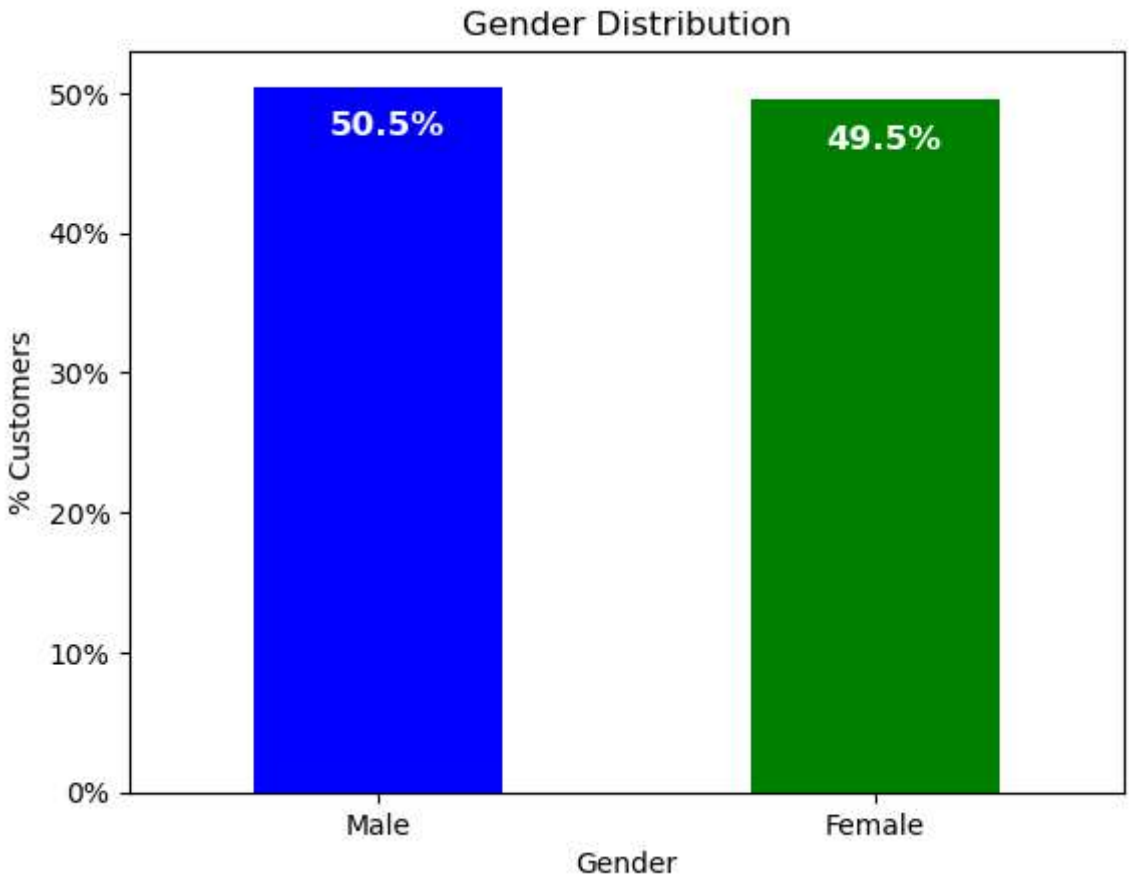
5 rows × 46 columns

In [8]:
```
colors=['orange','blue','red']
plt.figure(figsize=(15,8))
df_dummies.corr()['Churn'].sort_values(ascending = False).plot(kind='bar',colo
```

Out[8]: <Axes: >

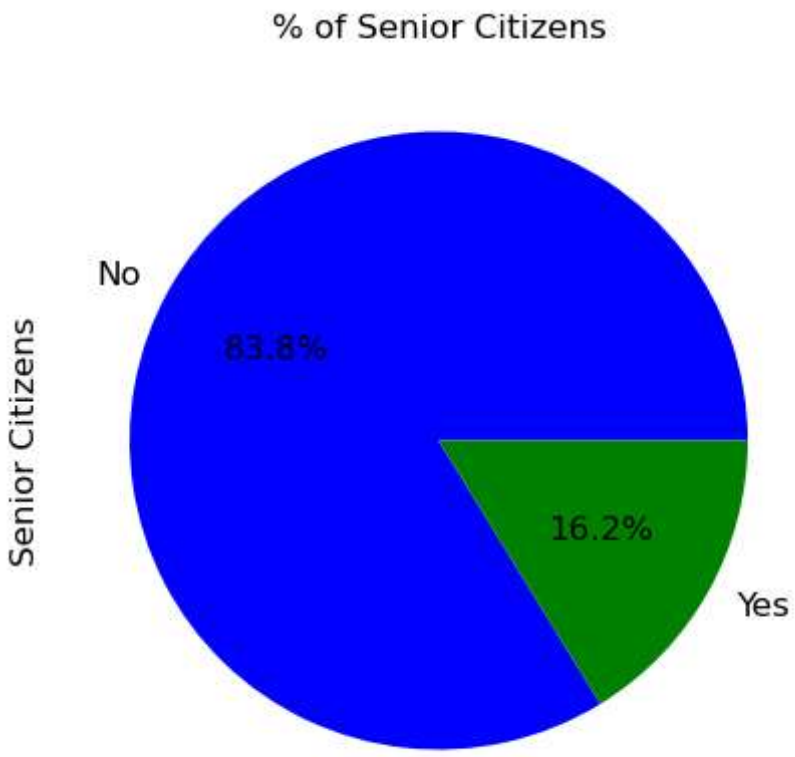


In [9]:
```
colors = ['blue','green']
ax = (telecom_cust['gender'].value_counts()*100.0 /len(telecom_cust)).plot(kir
                                                                           sta
                                                                          rot
                                                                          colc

ax.yaxis.set_major_formatter(mtick.PercentFormatter())
ax.set_ylabel('% Customers')
ax.set_xlabel('Gender')
ax.set_ylabel('% Customers')
ax.set_title('Gender Distribution')
totals=[]
for i in ax.patches:
    totals.append(i.get_width())
total=sum(totals)
for i in ax.patches:
    ax.text(i.get_x()+.15, i.get_height()-3.5, \
            str(round((i.get_height()/total), 1))+'%',
            fontsize=12,
            color='white',
           weight='bold')
```

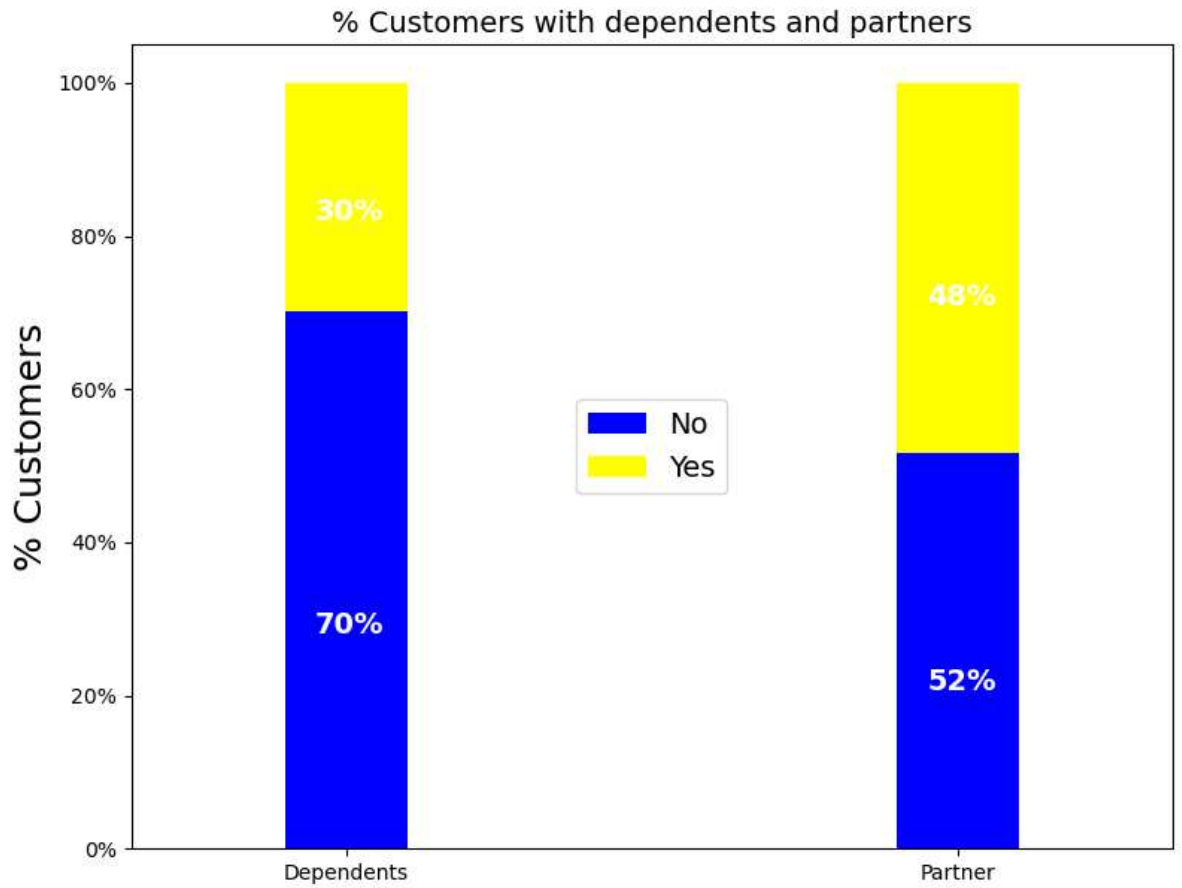


In [10]:
```
ax = (telecom_cust['SeniorCitizen'].value_counts()*100.0 /len(telecom_cust))\
.plot.pie(autopct='%.1f%%', labels = ['No','Yes'],figsize=(5,5),fontsize = 12,
ax.yaxis.set_major_formatter(mtick.PercentFormatter())
ax.set_ylabel('Senior Citizens',fontsize=12)
ax.set_title('% of Senior Citizens',fontsize=12)
```

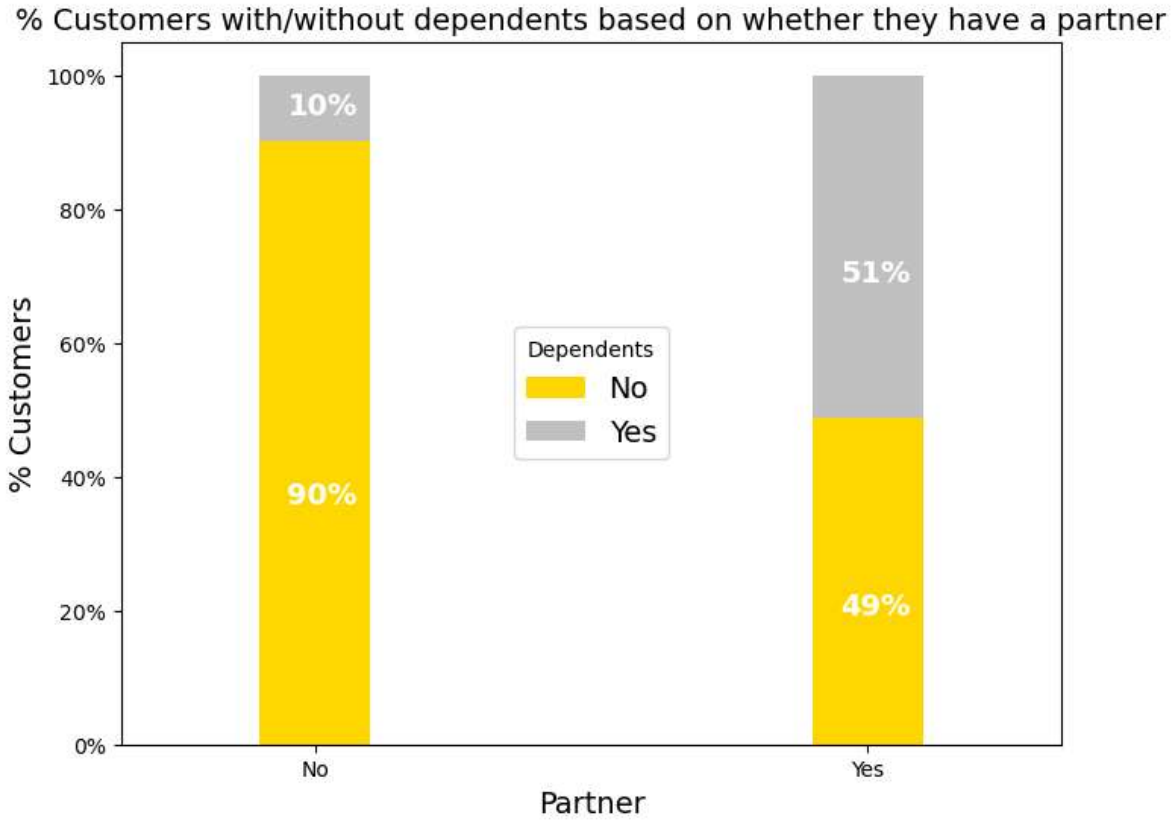
Out[10]:
```
Text(0.5, 1.0, '% of Senior Citizens')
```



In [11]:
```
df2=pd.melt(telecom_cust, id_vars=['customerID'], value_vars=['Dependents','Pa
df3=df2.groupby(['variable','value']).count().unstack()
df3=df3*100/len(telecom_cust)
colors=['blue','yellow']
ax=df3.loc[:,'customerID'].plot.bar(stacked=True, color=colors,
                                    figsize=(9,7),rot=0,
                                   width=0.2)
ax.yaxis.set_major_formatter(mtick.PercentFormatter())
ax.set_ylabel('% Customers',size = 18)
ax.set_xlabel('')
ax.set_title('% Customers with dependents and partners',size = 14)
ax.legend(loc='center',prop={'size':14})
for p in ax.patches:
    width,height=p.get_width(),p.get_height()
    x,y=p.get_xy()
    ax.annotate('{:.0f}%'.format(height), (p.get_x()+.25*width, p.get_y()+.4*h
                color='white',
               weight='bold',
               size=14)
```



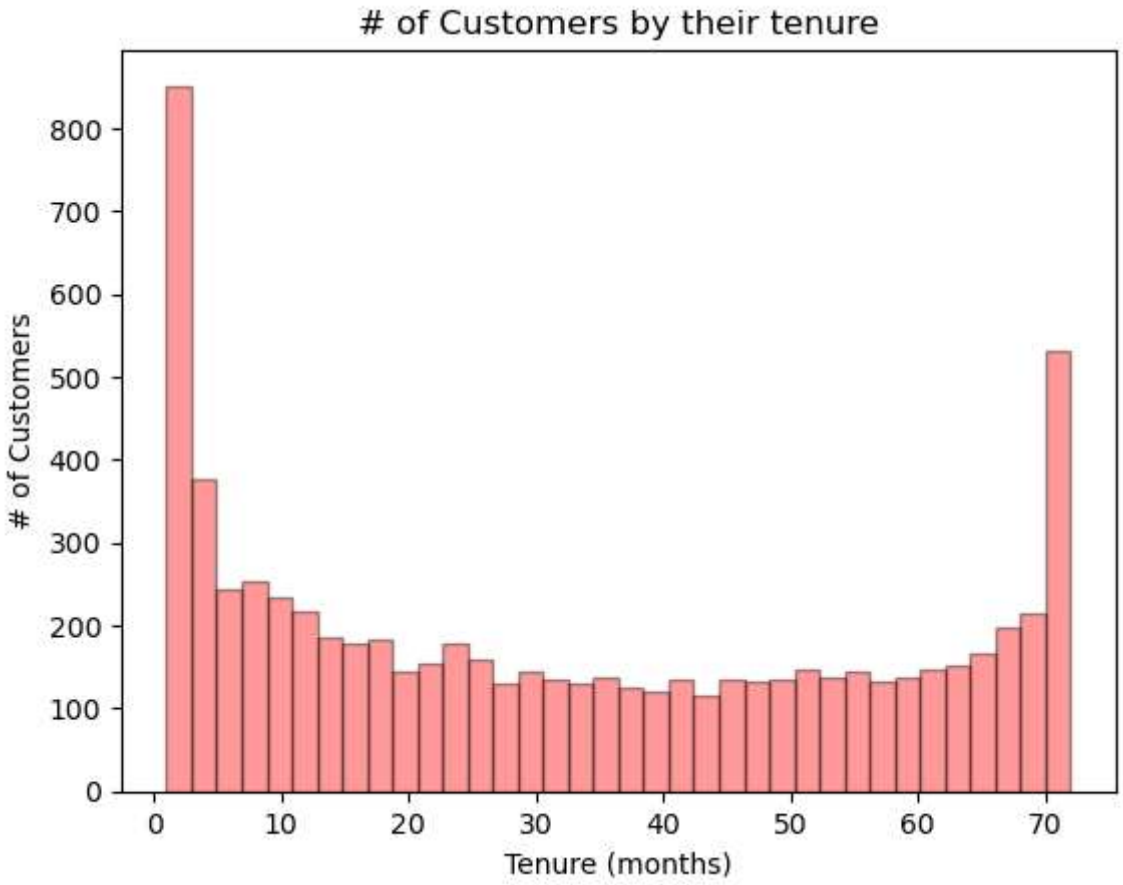
In [12]:
```
colors=['gold','silver']
partner_dependents=telecom_cust.groupby(['Partner','Dependents']).size().unsta
ax=(partner_dependents.T*100.0 / partner_dependents.T.sum()).T.plot(kind='bar'
                                                                    width=0.2,
                                                                    stacked=True,
                                                                    rot=0,
                                                                    figsize=(8,6),
                                                                    color=colors)
ax.yaxis.set_major_formatter(mtick.PercentFormatter())
ax.legend(loc='center',prop={'size':14},title ='Dependents',fontsize =14)
ax.set_ylabel('% Customers',size = 14)
ax.set_title('% Customers with/without dependents based on whether they have a
ax.xaxis.label.set_size(14)
for p in ax.patches:
    width, height = p.get_width(), p.get_height()
    x,y=p.get_xy()
    ax.annotate('{:.0f}%'.format(height), (p.get_x()+.25*width, p.get_y()+.4*h
                color='white',
               weight='bold',
               size=14)
```



In [38]:
```
ax=sns.distplot(telecom_cust['tenure'],hist=True, kde=False,
                bins=int(180/5), color = 'red',
                hist_kws={'edgecolor':'black'},
                kde_kws={'linewidth': 4})
ax.set_ylabel('# of Customers')
ax.set_xlabel('Tenure (months)')
ax.set_title('# of Customers by their tenure')
```

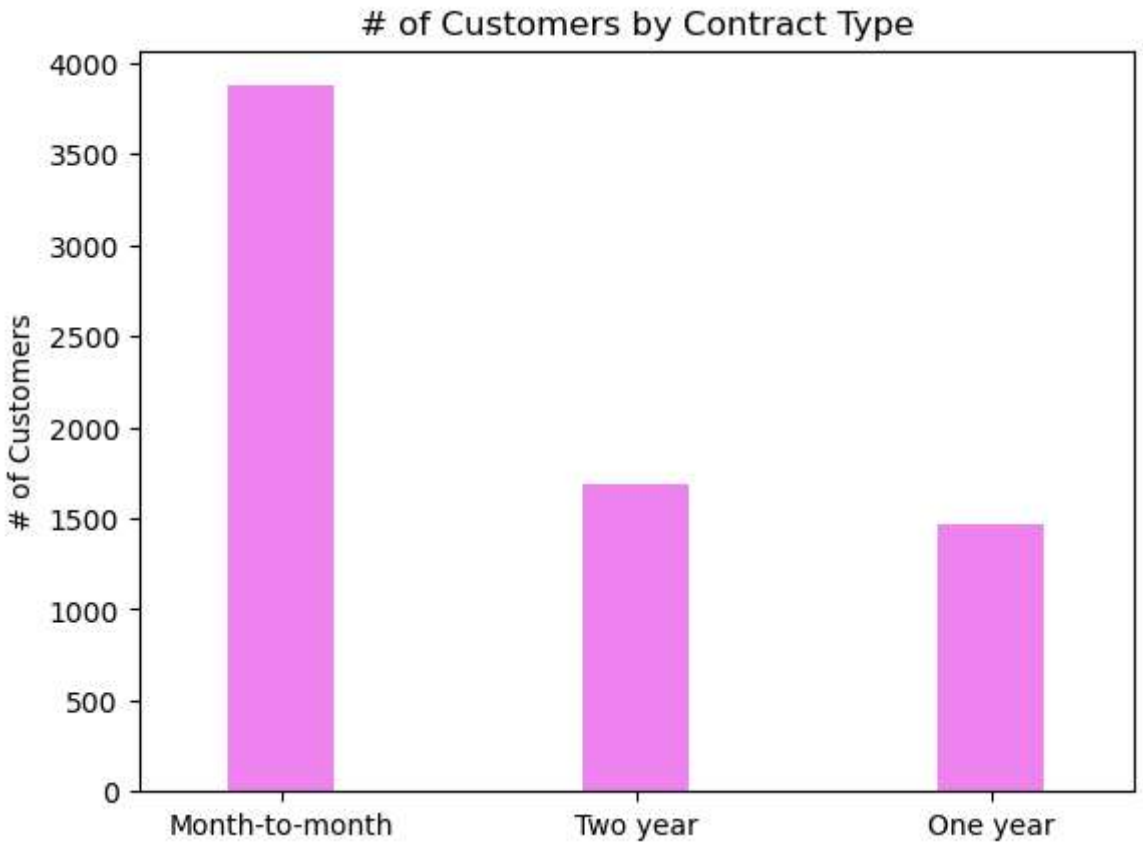
```
C:\Users\SRI KAAVYA\AppData\Local\Temp\ipykernel_7180\3753719227.py:1: UserWa
rning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751 (https://gis
t.github.com/mwaskom/de44147ed2974457ad6372750bbe5751)

  ax=sns.distplot(telecom_cust['tenure'],hist=True, kde=False,
```

Out[38]: Text(0.5, 1.0, '# of Customers by their tenure')



In [41]:
```
ax=telecom_cust['Contract'].value_counts().plot(kind='bar',rot=0,width=0.3,col
ax.set_ylabel('# of Customers')
ax.set_title('# of Customers by Contract Type')
```

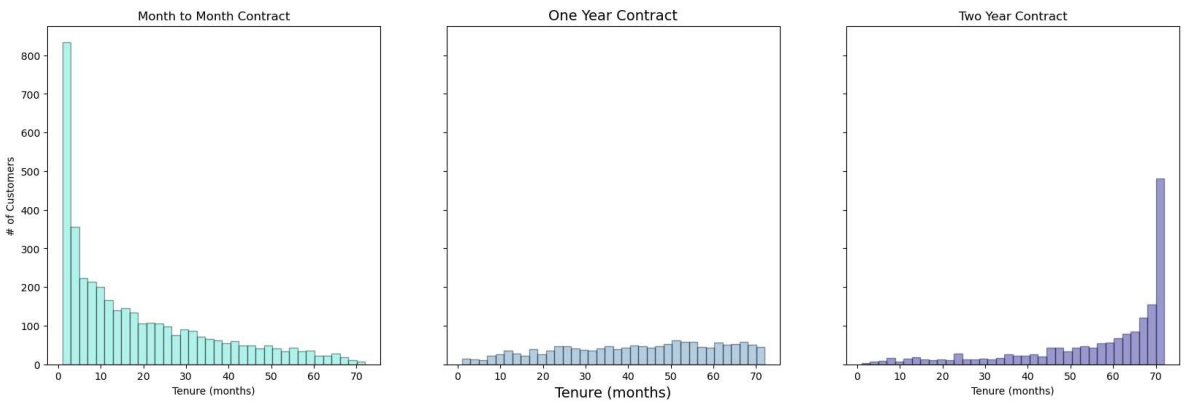
Out[41]:
```
Text(0.5, 1.0, '# of Customers by Contract Type')
```



In [15]:
```
fig,(ax1,ax2,ax3)=plt.subplots(nrows=1,ncols=3,sharey=True,figsize=(20,6))
ax = sns.distplot(telecom_cust[telecom_cust['Contract']=='Month-to-month']['te
                  hist=True, kde=False,
                  bins=int(180/5), color = 'turquoise',
                  hist_kws={'edgecolor':'black'},
                  kde_kws={'linewidth': 4},
                 ax=ax1)
ax.set_ylabel('# of Customers')
ax.set_xlabel('Tenure (months)')
ax.set_title('Month to Month Contract')

ax=sns.distplot(telecom_cust[telecom_cust['Contract']=='One year']['tenure'],
                  hist=True,kde=False,
                  bins=int(180/5), color = 'steelblue',
                  hist_kws={'edgecolor':'black'},
                  kde_kws={'linewidth': 4},
                 ax=ax2)
ax.set_xlabel('Tenure (months)',size = 14)
ax.set_title('One Year Contract',size = 14)

ax = sns.distplot(telecom_cust[telecom_cust['Contract']=='Two year']['tenure']
                  hist=True, kde=False,
                  bins=int(180/5), color = 'darkblue',
                  hist_kws={'edgecolor':'black'},
                  kde_kws={'linewidth': 4},
                 ax=ax3)
ax.set_xlabel('Tenure (months)')
ax.set_title('Two Year Contract')
```

```
C:\Users\SRI KAAVYA\AppData\Local\Temp\ipykernel_7180\1065370864.py:2: UserWa
rning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751 (https://gis
t.github.com/mwaskom/de44147ed2974457ad6372750bbe5751)

  ax = sns.distplot(telecom_cust[telecom_cust['Contract']=='Month-to-month']
['tenure'],
C:\Users\SRI KAAVYA\AppData\Local\Temp\ipykernel_7180\1065370864.py:12: UserW
arning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751 (https://gis
t.github.com/mwaskom/de44147ed2974457ad6372750bbe5751)

  ax=sns.distplot(telecom_cust[telecom_cust['Contract']=='One year']['tenur
e'],
C:\Users\SRI KAAVYA\AppData\Local\Temp\ipykernel_7180\1065370864.py:21: UserW
arning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751 (https://gis
t.github.com/mwaskom/de44147ed2974457ad6372750bbe5751)

  ax = sns.distplot(telecom_cust[telecom_cust['Contract']=='Two year']['tenur
e'],
```

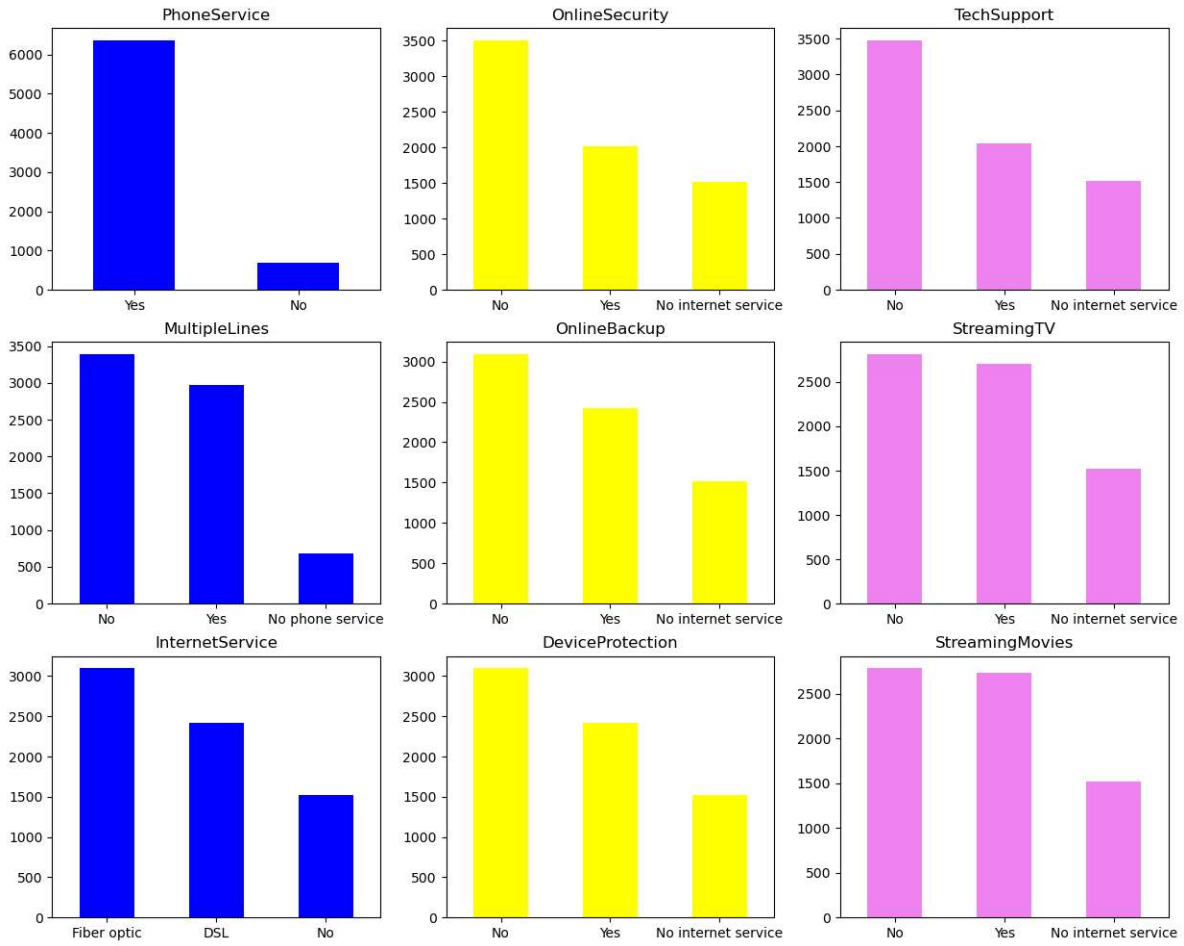
Out[15]: Text(0.5, 1.0, 'Two Year Contract')



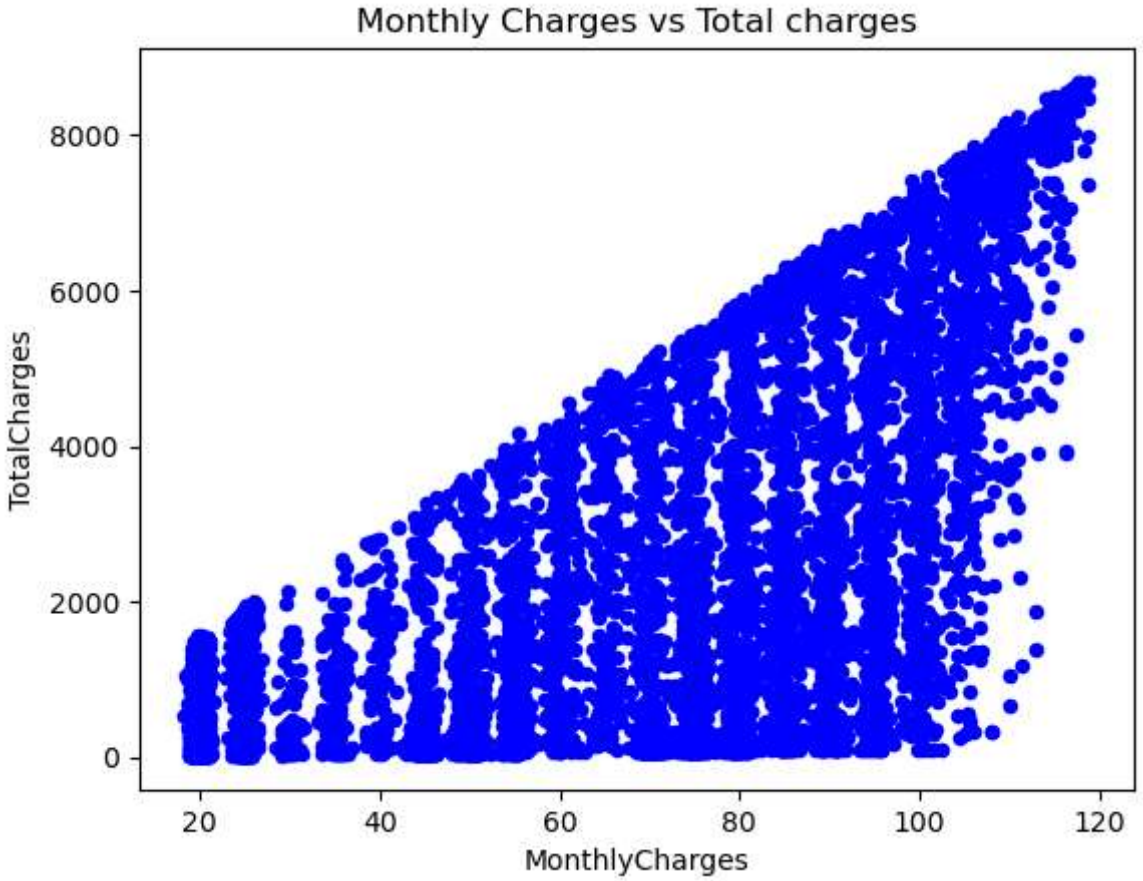
In [16]:
```
telecom_cust.columns.values
```

Out[16]:
```
array(['customerID', 'gender', 'SeniorCitizen', 'Partner', 'Dependents',
       'tenure', 'PhoneService', 'MultipleLines', 'InternetService',
       'OnlineSecurity', 'OnlineBackup', 'DeviceProtection',
       'TechSupport', 'StreamingTV', 'StreamingMovies', 'Contract',
       'PaperlessBilling', 'PaymentMethod', 'MonthlyCharges',
       'TotalCharges', 'Churn'], dtype=object)
```

In [46]:
```
services=['PhoneService','MultipleLines','InternetService','OnlineSecurity',
          'OnlineBackup','DeviceProtection','TechSupport','StreamingTV','Stre
fig,axes=plt.subplots(nrows=3,ncols=3,figsize=(15,12))
for i,item in enumerate(services):
    if i<3:
        ax=telecom_cust[item].value_counts().plot(kind='bar',ax=axes[i,0],rot=
    elif i>=3 and i < 6:
        ax = telecom_cust[item].value_counts().plot(kind = 'bar',ax=axes[i-3,1
    elif i < 9:
        ax = telecom_cust[item].value_counts().plot(kind = 'bar',ax=axes[i-6,2
    ax.set_title(item)
```



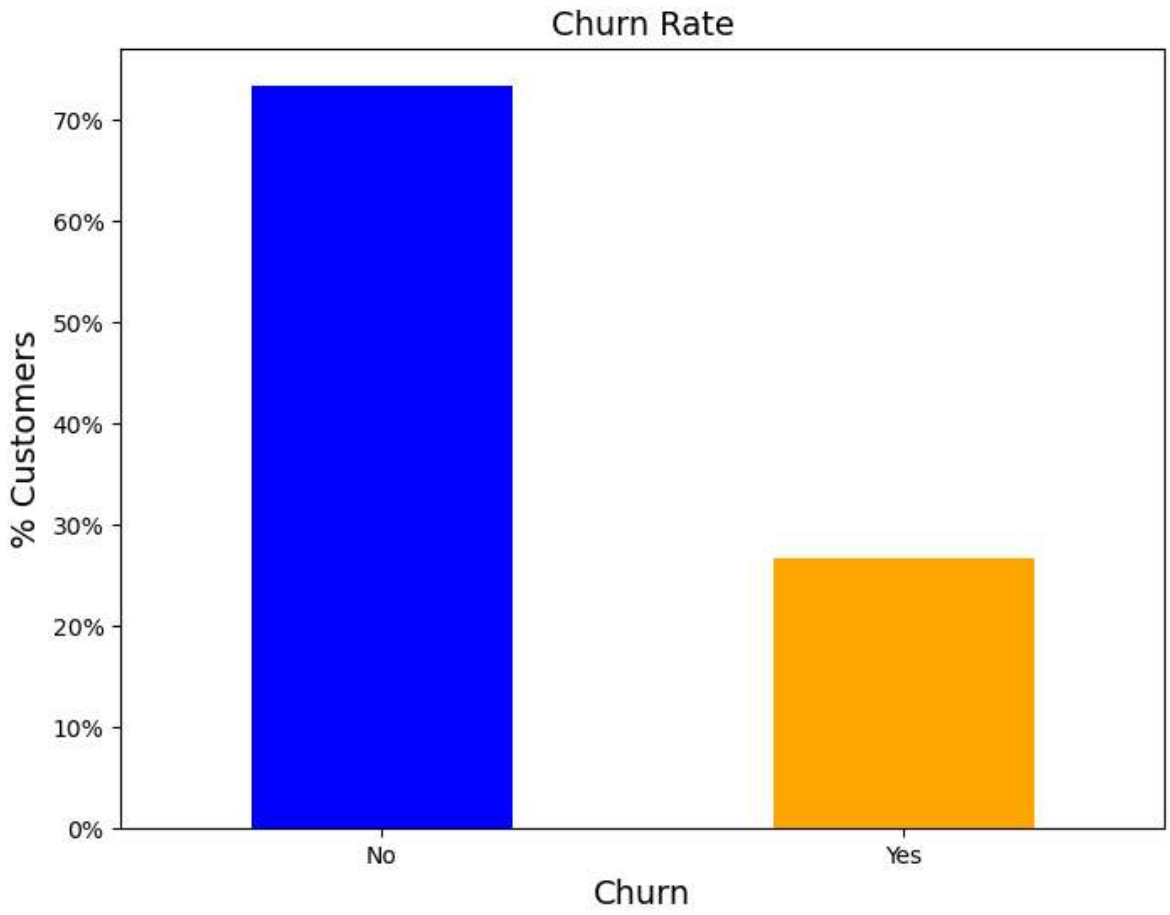
In [50]:
```
telecom_cust[['MonthlyCharges','TotalCharges']].plot.scatter(x = 'MonthlyCharg
                                                             y='TotalCharges'
plt.title('Monthly Charges vs Total charges')
plt.show()
```



In [51]:
```
colors = ['blue','orange']
ax = (telecom_cust['Churn'].value_counts()*100.0 /len(telecom_cust)).plot(kind
                                                                          sta
                                                                         rot
                                                                         colo
                                                                        figsi

ax.yaxis.set_major_formatter(mtick.PercentFormatter())
ax.set_ylabel('% Customers',size = 14)
ax.set_xlabel('Churn',size = 14)
ax.set_title('Churn Rate', size = 14)
totals = []
for i in ax.patches:
    totals.append(i.get_width())
total = sum(totals)
for i in ax.patches:
    ax.text(i.get_x()+.15, i.get_height()-4.0, \
            str(round((i.get_height()/total), 1))+'%',
            fontsize=12,
            color='white',
           weight = 'bold',
           size = 14)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[51], line 23
     20 total = sum(totals)
     22 for i in ax.patches:
---> 23     ax.text(i.get_x()+.15, i.get_height()-4.0, \
     24             str(round((i.get_height()/total), 1))+'%',
     25             fontsize=12,
     26             color='white',
     27            weight = 'bold',
     28            size = 14)

File ~\anaconda3\Lib\site-packages\matplotlib\axes\_axes.py:689, in Axes.text
(self, x, y, s, fontdict, **kwargs)
    628 """
    629 Add text to the Axes.
    630
   (...)
    679     >>> text(x, y, s, bbox=dict(facecolor='red', alpha=0.5))
    680 """
    681 effective_kwargs = {
    682     'verticalalignment': 'baseline',
    683     'horizontalalignment': 'left',
   (...)
    687     **kwargs,
    688 }
--> 689 t = mtext.Text(x, y, text=s, **effective_kwargs)
    690 t.set_clip_path(self.patch)
    691 self._add_text(t)

File ~\anaconda3\Lib\site-packages\matplotlib\_api\deprecation.py:454, in mak
e_keyword_only.<locals>.wrapper(*args, **kwargs)
    448 if len(args) > name_idx:
    449     warn_deprecated(
    450         since, message="Passing the %(name)s %(obj_type)s "
    451         "positionally is deprecated since Matplotlib %(since)s; the "
    452         "parameter will become keyword-only %(removal)s.",
    453         name=name, obj_type=f"parameter of {func.__name__}()")
--> 454 return func(*args, **kwargs)

File ~\anaconda3\Lib\site-packages\matplotlib\text.py:183, in Text.__init__(s
elf, x, y, text, color, verticalalignment, horizontalalignment, multialignmen
t, fontproperties, rotation, linespacing, rotation_mode, usetex, wrap, transf
orm_rotates_text, parse_math, **kwargs)
    167 self._text = ''
    168 self._reset_visual_defaults(
    169     text=text,
    170     color=color,
   (...)
    181     rotation_mode=rotation_mode,
    182 )
--> 183 self.update(kwargs)

File ~\anaconda3\Lib\site-packages\matplotlib\text.py:223, in Text.update(sel
f, kwargs)
    221 def update(self, kwargs):
    222     # docstring inherited
```

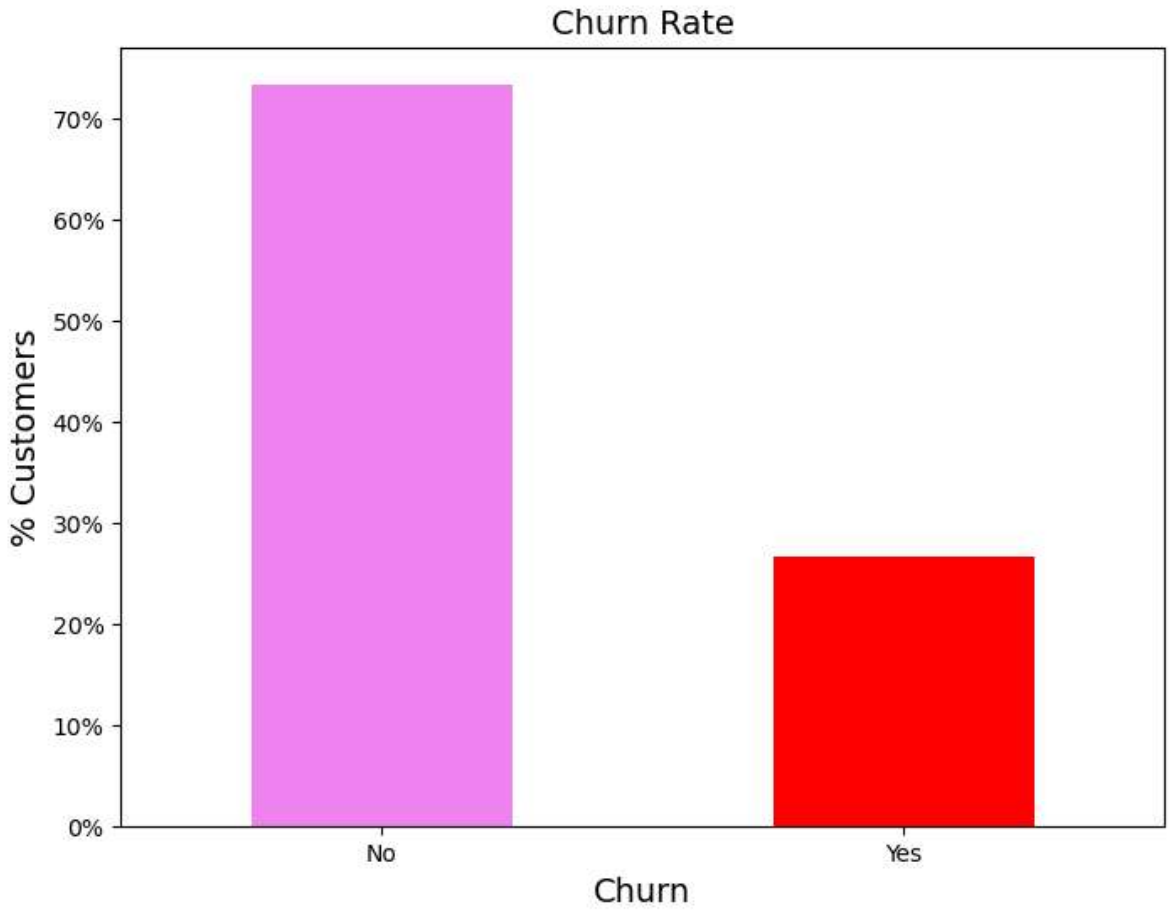
```
--> 223     kwargs = cbook.normalize_kwargs(kwargs, Text)
    224     sentinel = object()  # bbox can be None, so use another sentinel.
    225     # Update fontproperties first, as it has lowest priority.

File ~\anaconda3\Lib\site-packages\matplotlib\cbook\__init__.py:1774, in normalize_kwargs(kw, alias_mapping)
   1772 canonical = to_canonical.get(k, k)
   1773 if canonical in canonical_to_seen:
-> 1774     raise TypeError(f"Got both {canonical_to_seen[canonical]!r} and "
   1775                     f"{k!r}, which are aliases of one another")
   1776 canonical_to_seen[canonical] = k
   1777 ret[canonical] = v

TypeError: Got both 'fontsize' and 'size', which are aliases of one another
```



In [52]:
```
import pandas as pd
import matplotlib.pyplot as plt
import matplotlib.ticker as mtick
colors = ['violet', 'red']

churn_percent = telecom_cust['Churn'].value_counts() * 100.0 / len(telecom_cus

ax = churn_percent.plot(kind='bar', stacked=True, rot=0, color=colors, figsize
ax.yaxis.set_major_formatter(mtick.PercentFormatter())
ax.set_ylabel('% Customers', size=14)
ax.set_xlabel('Churn', size=14)
ax.set_title('Churn Rate', size=14)

for i in ax.patches:
    percentage = f"{round(i.get_height(), 1)}%"
    ax.text(i.get_x() + 0.15, i.get_height() - 4.0, percentage, fontsize=12, c

plt.show()
```

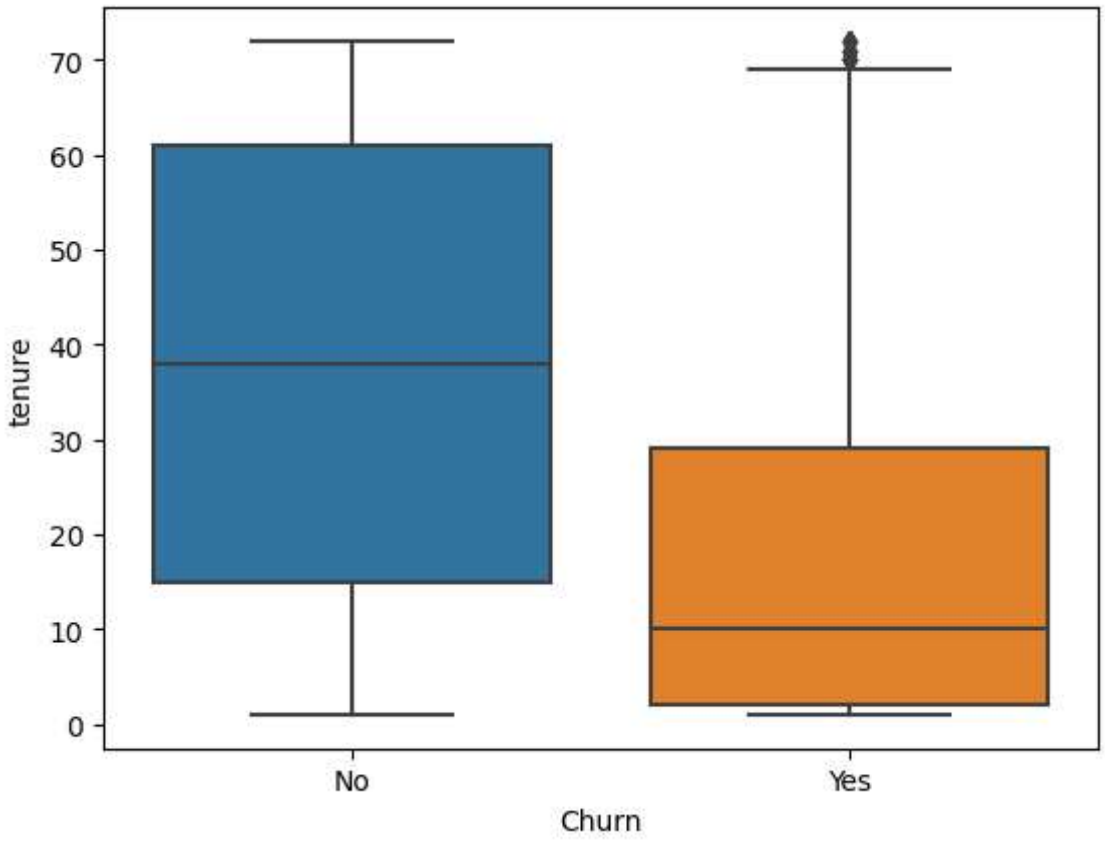
```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[52], line 16
     14 for i in ax.patches:
     15     percentage = f"{round(i.get_height(), 1)}%"
---> 16     ax.text(i.get_x() + 0.15, i.get_height() - 4.0, percentage, fonts
ize=12, color='white', weight='bold', size=14)
     18 plt.show()

File ~\anaconda3\Lib\site-packages\matplotlib\axes\_axes.py:689, in Axes.text
(self, x, y, s, fontdict, **kwargs)
    628 """
    629 Add text to the Axes.
    630
   (...)
    679     >>> text(x, y, s, bbox=dict(facecolor='red', alpha=0.5))
    680 """
    681 effective_kwargs = {
    682     'verticalalignment': 'baseline',
    683     'horizontalalignment': 'left',
   (...)
    687     **kwargs,
    688 }
--> 689 t = mtext.Text(x, y, text=s, **effective_kwargs)
    690 t.set_clip_path(self.patch)
    691 self._add_text(t)

File ~\anaconda3\Lib\site-packages\matplotlib\_api\deprecation.py:454, in mak
e_keyword_only.<locals>.wrapper(*args, **kwargs)
    448 if len(args) > name_idx:
    449     warn_deprecated(
    450         since, message="Passing the %(name)s %(obj_type)s "
    451         "positionally is deprecated since Matplotlib %(since)s; the "
    452         "parameter will become keyword-only %(removal)s.",
    453         name=name, obj_type=f"parameter of {func.__name__}()")
--> 454 return func(*args, **kwargs)

File ~\anaconda3\Lib\site-packages\matplotlib\text.py:183, in Text.__init__(s
elf, x, y, text, color, verticalalignment, horizontalalignment, multialignmen
t, fontproperties, rotation, linespacing, rotation_mode, usetex, wrap, transf
orm_rotates_text, parse_math, **kwargs)
    167 self._text = ''
    168 self._reset_visual_defaults(
    169     text=text,
    170     color=color,
   (...)
    181     rotation_mode=rotation_mode,
    182 )
--> 183 self.update(kwargs)

File ~\anaconda3\Lib\site-packages\matplotlib\text.py:223, in Text.update(sel
f, kwargs)
    221 def update(self, kwargs):
    222     # docstring inherited
--> 223     kwargs = cbook.normalize_kwargs(kwargs, Text)
    224     sentinel = object()  # bbox can be None, so use another sentinel.
    225     # Update fontproperties first, as it has lowest priority.
```

```
File ~\anaconda3\Lib\site-packages\matplotlib\cbook\__init__.py:1774, in normalize_kwargs(kw, alias_mapping)
   1772 canonical = to_canonical.get(k, k)
   1773 if canonical in canonical_to_seen:
-> 1774     raise TypeError(f"Got both {canonical_to_seen[canonical]!r} and "
   1775                     f"{k!r}, which are aliases of one another")
   1776 canonical_to_seen[canonical] = k
   1777 ret[canonical] = v

TypeError: Got both 'fontsize' and 'size', which are aliases of one another
```

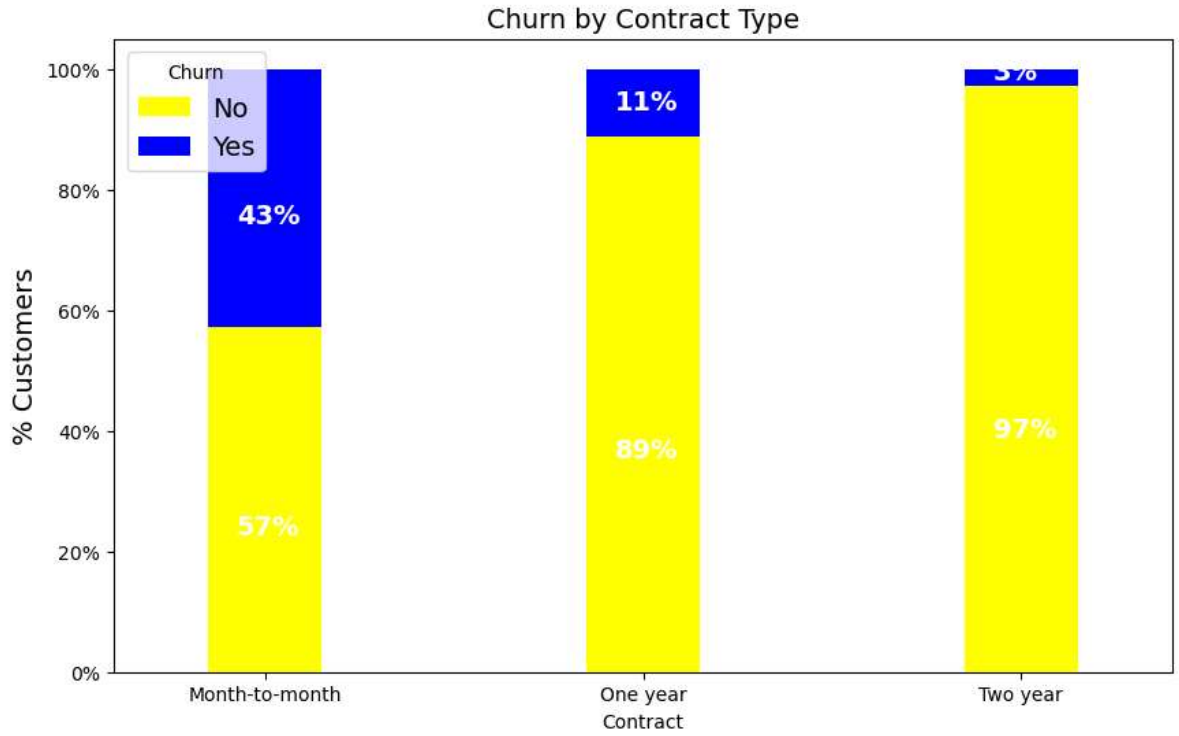


In [21]:
```
sns.boxplot(x = telecom_cust.Churn, y = telecom_cust.tenure)
```

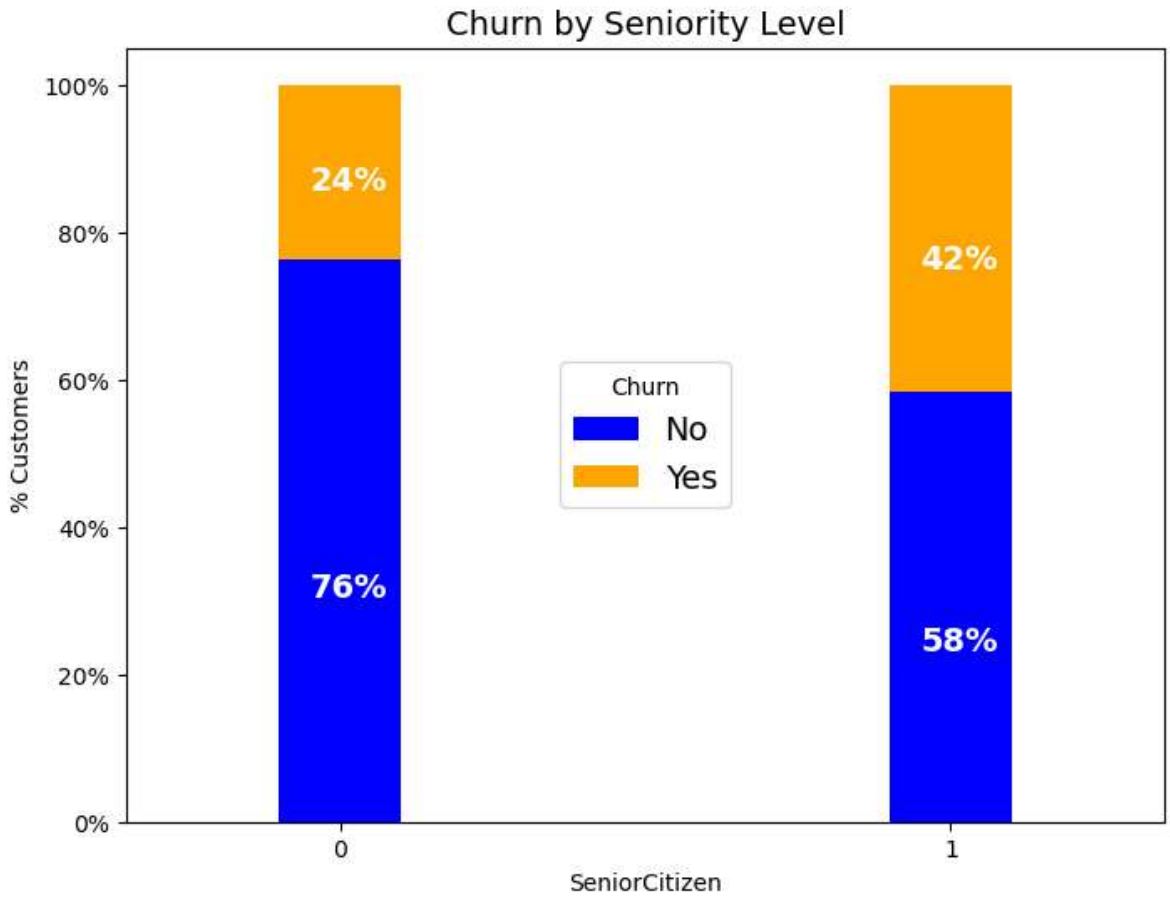
Out[21]:
```
<Axes: xlabel='Churn', ylabel='tenure'>
```



In [53]:
```
colors = ['yellow','blue']
contract_churn = telecom_cust.groupby(['Contract','Churn']).size().unstack()

ax = (contract_churn.T*100.0 / contract_churn.T.sum()).T.plot(kind='bar',
                                                              width = 0.3,
                                                              stacked = True
                                                              rot = 0,
                                                              figsize = (10,
                                                              color = colors
ax.yaxis.set_major_formatter(mtick.PercentFormatter())
ax.legend(loc='best',prop={'size':14},title = 'Churn')
ax.set_ylabel('% Customers',size = 14)
ax.set_title('Churn by Contract Type',size = 14)
for p in ax.patches:
    width, height = p.get_width(), p.get_height()
    x, y = p.get_xy()
    ax.annotate('{:.0f}%'.format(height), (p.get_x()+.25*width, p.get_y()+.4*h
                color = 'white',
               weight = 'bold',
               size = 14)
```



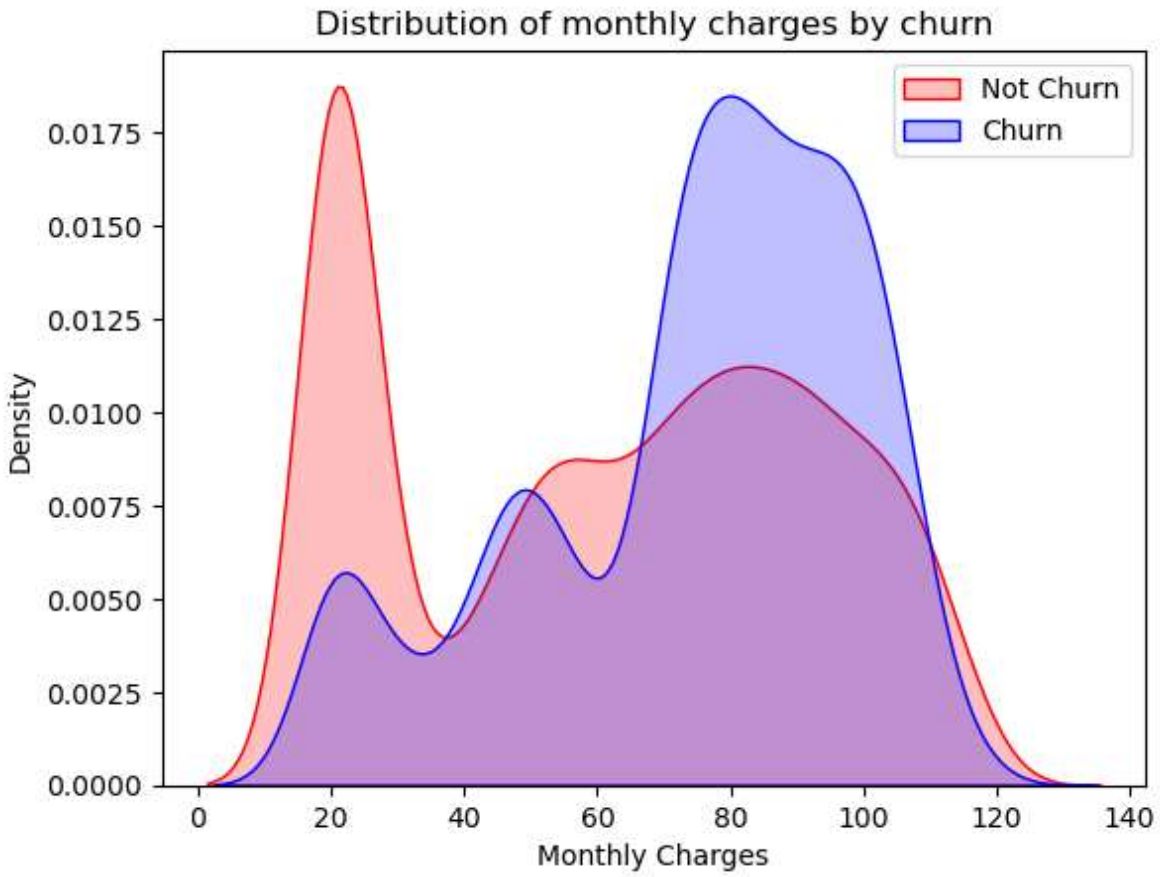
In [54]:
```
colors = ['blue','']
seniority_churn = telecom_cust.groupby(['SeniorCitizen','Churn']).size().unsta

ax = (seniority_churn.T*100.0 / seniority_churn.T.sum()).T.plot(kind='bar',
                                                                width = 0.2,
                                                                stacked = True
                                                                rot = 0,
                                                                figsize = (8,6
                                                                color = colors
ax.yaxis.set_major_formatter(mtick.PercentFormatter())
ax.legend(loc='center',prop={'size':14},title = 'Churn')
ax.set_ylabel('% Customers')
ax.set_title('Churn by Seniority Level',size = 14)
for p in ax.patches:
    width, height = p.get_width(), p.get_height()
    x, y = p.get_xy()
    ax.annotate('{:.0f}%'.format(height), (p.get_x()+.25*width, p.get_y()+.4*h
                color = 'white',
               weight = 'bold',size =14)
```



In [24]:
```
ax = sns.kdeplot(telecom_cust.MonthlyCharges[(telecom_cust["Churn"] == 'No') ],
                 color="Red", shade = True)
ax = sns.kdeplot(telecom_cust.MonthlyCharges[(telecom_cust["Churn"] == 'Yes') ],
                 ax =ax, color="Blue", shade= True)
ax.legend(["Not Churn","Churn"],loc='upper right')
ax.set_ylabel('Density')
ax.set_xlabel('Monthly Charges')
ax.set_title('Distribution of monthly charges by churn')
```

```
C:\Users\SRI KAAVYA\AppData\Local\Temp\ipykernel_7180\2862132292.py:1: Future
Warning:

`shade` is now deprecated in favor of `fill`; setting `fill=True`.
This will become an error in seaborn v0.14.0; please update your code.

  ax = sns.kdeplot(telecom_cust.MonthlyCharges[(telecom_cust["Churn"] == 'N
o') ],
C:\Users\SRI KAAVYA\AppData\Local\Temp\ipykernel_7180\2862132292.py:3: Future
Warning:

`shade` is now deprecated in favor of `fill`; setting `fill=True`.
This will become an error in seaborn v0.14.0; please update your code.

  ax = sns.kdeplot(telecom_cust.MonthlyCharges[(telecom_cust["Churn"] == 'Ye
s') ],
```

Out[24]: Text(0.5, 1.0, 'Distribution of monthly charges by churn')

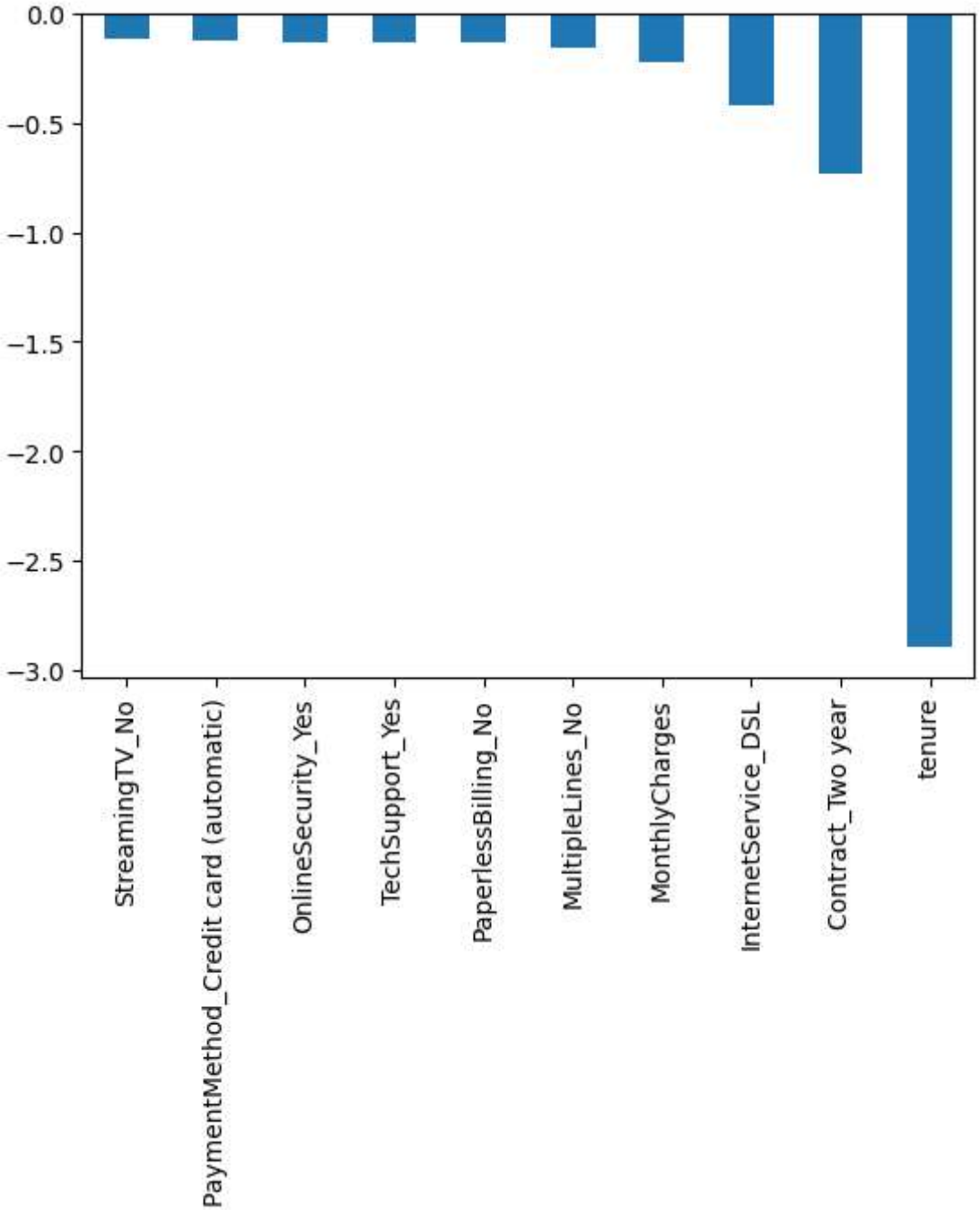


In [25]:
```
y = df_dummies['Churn'].values
X = df_dummies.drop(columns = ['Churn'])
from sklearn.preprocessing import MinMaxScaler
features = X.columns.values
scaler = MinMaxScaler(feature_range = (0,1))
scaler.fit(X)
X = pd.DataFrame(scaler.transform(X))
X.columns = features
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, randc
from sklearn.linear_model import LogisticRegression
model = LogisticRegression()
result = model.fit(X_train, y_train)
from sklearn import metrics
prediction_test = model.predict(X_test)
print (metrics.accuracy_score(y_test, prediction_test))
```

```
0.8075829383886256
```

In [27]:
```
print(weights.sort_values(ascending = False)[-10:].plot(kind='bar'))
```

```
Axes(0.125,0.11;0.775x0.77)
```

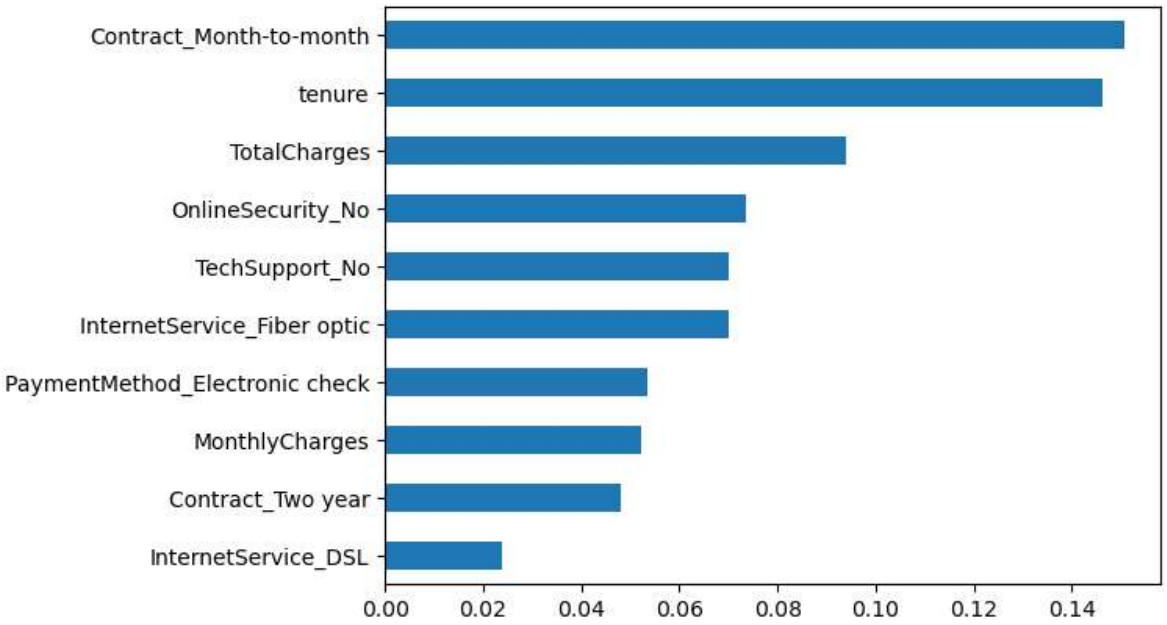


In [28]:
```
from sklearn.ensemble import RandomForestClassifier
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, randc
model_rf = RandomForestClassifier(n_estimators=1000 , oob_score = True, n_jobs
                                  random_state =50, max_features = "auto",
                                  max_leaf_nodes = 30)
model_rf.fit(X_train, y_train)
prediction_test = model_rf.predict(X_test)
print (metrics.accuracy_score(y_test, prediction_test))
```

```
C:\Users\SRI KAAVYA\anaconda3\Lib\site-packages\sklearn\ensemble\_forest.py:4
24: FutureWarning: `max_features='auto'` has been deprecated in 1.1 and will
be removed in 1.3. To keep the past behaviour, explicitly set `max_features
='sqrt'` or remove this parameter as it is also the default value for RandomF
orestClassifiers and ExtraTreesClassifiers.
  warn(

0.8088130774697939
```

In [29]:
```
importances = model_rf.feature_importances_
weights = pd.Series(importances,
                    index=X.columns.values)
weights.sort_values()[-10:].plot(kind = 'barh')
```

Out[29]: <Axes: >



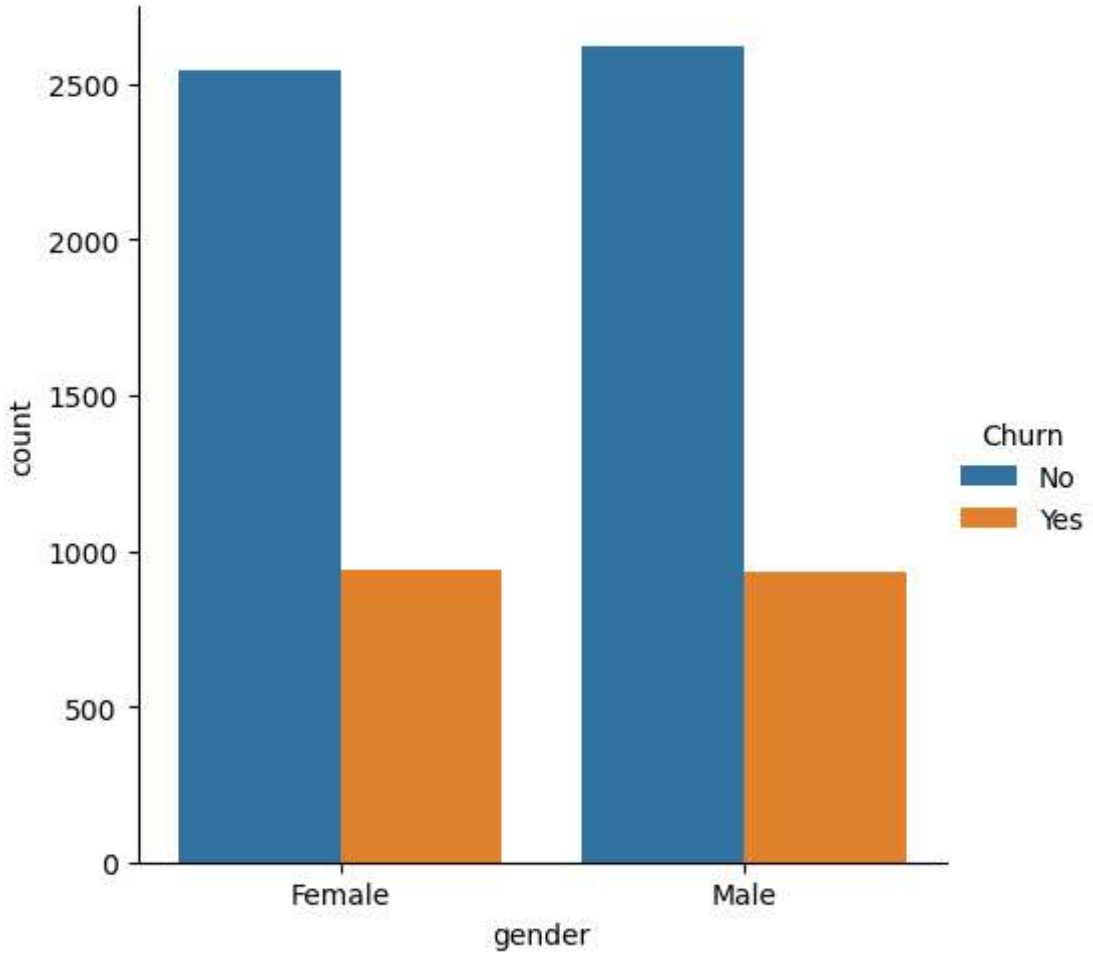
In [30]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, randc
from sklearn.svm import SVC

model.svm = SVC(kernel='linear')
model.svm.fit(X_train,y_train)
preds = model.svm.predict(X_test)
metrics.accuracy_score(y_test, preds)
```

Out[30]: 0.820184790334044

In [31]:
```
from sklearn.metrics import classification_report, confusion_matrix
print(confusion_matrix(y_test,preds))
```

```
[[953  89]
 [164 201]]
```

In [32]:
```
ax1 = sns.catplot(x="gender", kind="count", hue="Churn", data=telecom_cust,
                  estimator=lambda x: sum(x==0)*100.0/len(x))
```



In [36]:
```
from sklearn.ensemble import AdaBoostClassifier
model = AdaBoostClassifier()
model.fit(X_train,y_train)
preds = model.predict(X_test)
metrics.accuracy_score(y_test, preds)
```

Out[36]: 0.8159203980099502

In [35]:
```
from xgboost import XGBClassifier
model = XGBClassifier()
model.fit(X_train, y_train)
preds = model.predict(X_test)
metrics.accuracy_score(y_test, preds)
```

Out[35]: 0.8095238095238095

In [ ]: