Thinkful Supervised Learning Capstone: Dan McHenry

Telecom Customer Churn from Kaggle.com

Objective:

To build various models in order to predict whether a customer will drop telecommunications services (churn) or maintain those services based on various demographic and service specific data.

Original Feature Set:

String (alpha-numeric): customerID

Categorical: gender, SeniorCitizen, Partner, Dependents, PhoneService, MuiltipleLines, InternetService, OnlineSecurity, OnlineBackup, DeviceProtection, TechSupport, StreamingTV, StreamingMovies, Contract, PaperlessBilling, PaymentMethod, Churn

Continuous: tenure, MonthlyCharges, TotalCharges

Exploration

Data type issues:

```
In [4]:
1 # Check data types.
2 churn.dtypes
```

```
Out[4]:
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 [5]:
1 # View the size of the data.
2 churn.shape
```

```
Out[5]: (7043, 21)
```

In [4]:
```
# Check data types.
churn.dtypes
```

Out[4]:
```
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 [5]:
```
# View the size of the data.
churn.shape
```

Out[5]:
```
(7043, 21)
```

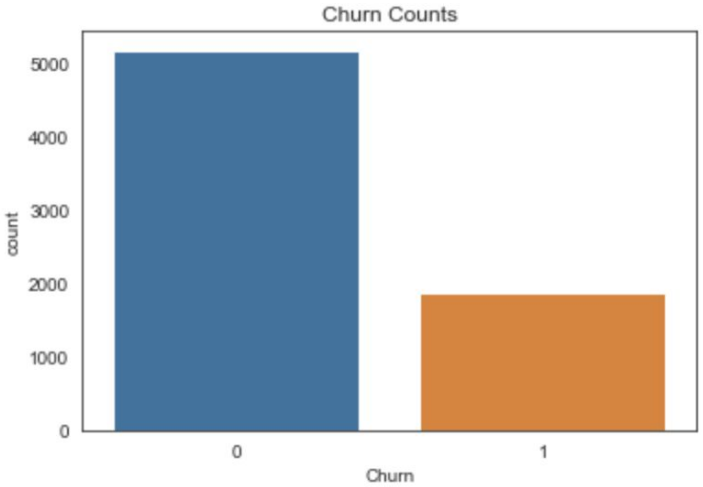
In [16]:
```
# Recheck data types.
churn_dum.dtypes
```

Out[16]:
```
customerID                        object
gender                             int64
SeniorCitizen                      int64
Partner                            int64
Dependents                         int64
tenure                             int64
PhoneService                       int64
MultipleLines                      int64
InternetService                    int64
OnlineSecurity                     int64
OnlineBackup                       int64
DeviceProtection                   int64
TechSupport                        int64
StreamingTV                        int64
StreamingMovies                    int64
Contract                           int64
PaperlessBilling                   int64
PaymentMethod                     object
MonthlyCharges                   float64
TotalCharges                      object
Churn                              int64
Bank transfer (automatic)          uint8
Credit card (automatic)            uint8
Electronic check                   uint8
Mailed check                       uint8
dtype: object
```

Exploration

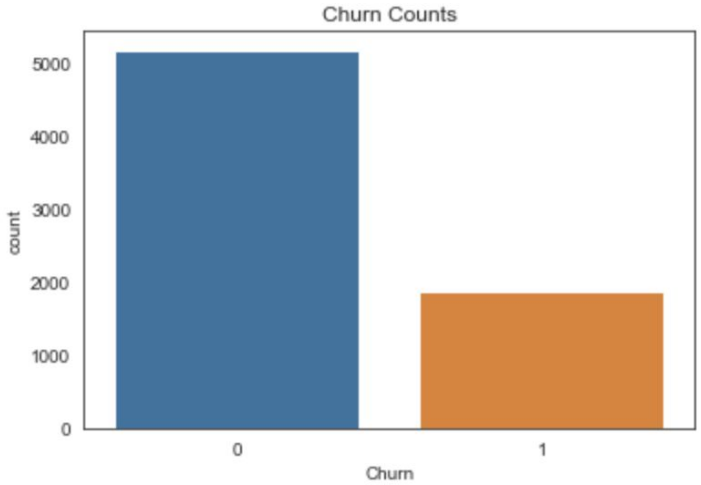
Value Counts

In [8]:
```
# View value counts.
sns.countplot('Churn', data=churn)
plt.title('Churn Counts')
plt.show()
pd.value_counts(churn['Churn'])
```



Out[8]:
```
0    5174
1    1869
Name: Churn, dtype: int64
```

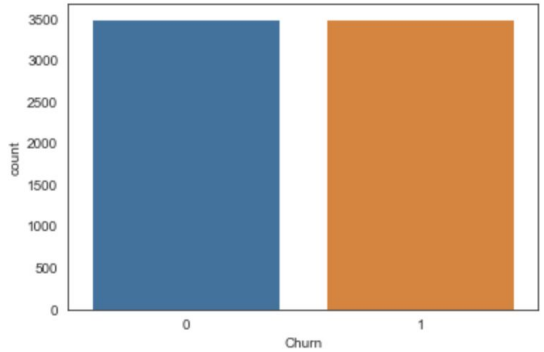
In [8]:
```
# View value counts.
sns.countplot('Churn', data=churn)
plt.title('Churn Counts')
plt.show()
pd.value_counts(churn['Churn'])
```



Out[8]:
```
0    5174
1    1869
Name: Churn, dtype: int64
```

```
from sklearn.utils import resample

# Upsample minority class.
churn_yes = resample(churn_yes,
                     replace=True,
                     n_samples=3500,
                     random_state=15)

# Downsample majority class.
churn_no = resample(churn_no,
                    replace=True,
                    n_samples=3500,
                    random_state=15)

# Combine upsampled minority class and downsampled majority class.
churn_sampled = pd.concat([churn_yes, churn_no])

# Display new class counts.
sns.countplot('Churn', data=churn_sampled)
churn_sampled.Churn.value_counts()
```

Out[24]:
```
1    3500
0    3500
Name: Churn, dtype: int64
```



Exploration

Missing Values

```
In [9]:  1  # Check for null values.
         2  churn.isnull().sum()

Out[9]:  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        0
         Churn               0
         dtype: int64
```

Exploration

Eleven values had a space (‘ ‘) in place of a value (as if someone had taken a value and hit the space bar to clear it out).

MonthlyCharges	TotalCharges	Churn
52.550		0
20.250		0
80.850		0
25.750		0
56.050		0
19.850		0
25.350		0
20.000		0
19.700		0
73.350		0

Exploration

PaymentMethod column needed to be dummy coded.

PaperlessBilling	PaymentMethod	MonthlyCharges
1	Electronic check	29.850
0	Mailed check	56.950
1	Mailed check	53.850
0	Bank transfer (automatic)	42.300
1	Electronic check	70.700

PaperlessBilling	PaymentMethod	MonthlyCharges
1	Electronic check	29.850
0	Mailed check	56.950
1	Mailed check	53.850
0	Bank transfer (automatic)	42.300
1	Electronic check	70.700

Bank transfer (automatic)	Credit card (automatic)	Electronic check	Mailed check
0	0	1	0
0	0	0	1
0	0	0	1
1	0	0	0
0	0	1	0

Exploration

Dummy code types had to be changed from uint8 to int64.

```
Bank transfer (automatic)      uint8
Credit card (automatic)        uint8
Electronic check               uint8
Mailed check                   uint8
```

```
Bank transfer (automatic)      int64
Credit card (automatic)        int64
Electronic check               int64
Mailed check                   int64
```

Run models

```
Naive Bayes Averaged Cross-Validation Scores: 76.06%.
K-Nearest Neighbors Averaged Cross-Validation Scores: 75.71%.
Decision Tree Averaged Cross-Validation Scores: 61.50%.
Random Forest Averaged Cross-Validation Scores: 90.29%.
Logistic Regression Averaged Cross-Validation Scores: 76.73%.
Linear Support Vector Averaged Cross-Validation Score: 71.09%.
Gradient Boosting Averaged Cross-Validation Score: 79.71%.
```

Run models

Rerun results included Scaling and PCA, but made minimal difference.

```
Naive Bayes PCA Averaged Cross-Validation Scores: 74.84%.
K-Nearest Neighbors PCA Averaged Cross-Validation Scores: 77.86%.
Decision Tree PCA Averaged Cross-Validation Scores: 64.29%.
Random Forest PCA Averaged Cross-Validation Scores: 90.31%.
Logistic Regression PCA Averaged Cross-Validation Scores: 75.79%.
Linear Support Vector PCA Averaged Cross-Validation Score: 75.86%.
Gradient Boosting PCA Averaged Cross-Validation Score: 79.04%.
```