In [60]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from scipy.stats import zscore
from category_encoders import OneHotEncoder
from sklearn.feature_selection import SelectKBest, f_regression
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_absolute_error, mean_squared_error
from sklearn.compose import ColumnTransformer
from sklearn.preprocessing import OneHotEncoder
```

In [43]:
```
# Importing the insurance dataset
insurance = pd.read_csv('C:\\Users\\EagleCORS\OneDrive\\Desktop\\David_assignment 7-8\\ADS-Assignment-7-8\\insurance.csv')
```

In [44]:
```
# Explore the data using pandas exploratory tools
print(insurance.head())
```

```
   age     sex     bmi  children smoker     region      charges
0   19  female  27.900         0    yes  southwest  16884.92400
1   18    male  33.770         1     no  southeast   1725.55230
2   28    male  33.000         3     no  southeast   4449.46200
3   33    male  22.705         0     no  northwest  21984.47061
4   32    male  28.880         0     no  northwest   3866.85520
```

In [45]:
```
print(insurance.describe())
```

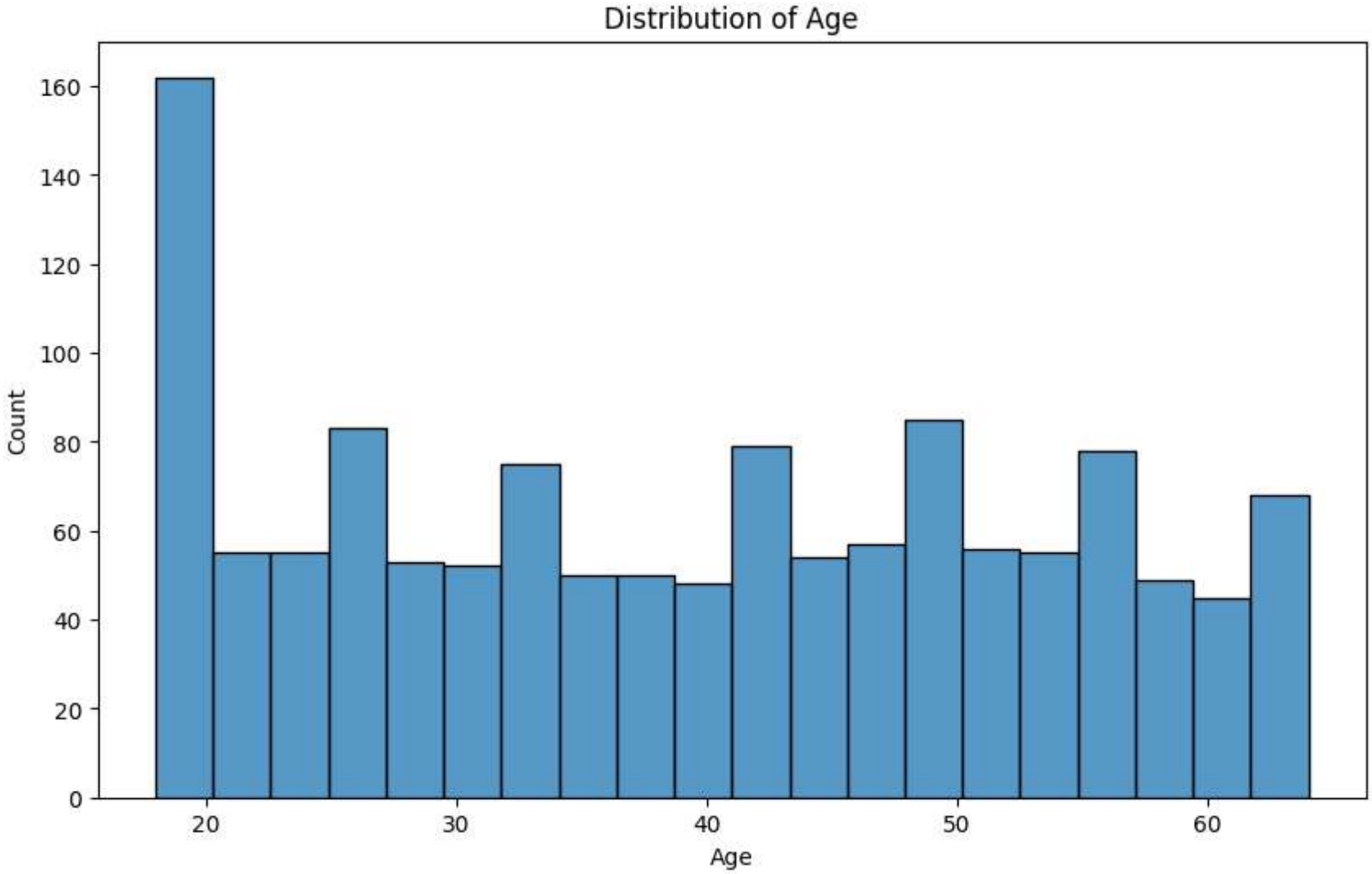
```
               age          bmi     children       charges
count  1338.000000  1338.000000  1338.000000   1338.000000
mean     39.207025    30.663397     1.094918  13270.422265
std      14.049960     6.098187     1.205493  12110.011237
min      18.000000    15.960000     0.000000   1121.873900
25%      27.000000    26.296250     0.000000   4740.287150
50%      39.000000    30.400000     1.000000   9382.033000
75%      51.000000    34.693750     2.000000  16639.912515
max      64.000000    53.130000     5.000000  63770.428010
```

In [46]:
```
print(insurance.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1338 entries, 0 to 1337
Data columns (total 7 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   age       1338 non-null   int64  
 1   sex       1338 non-null   object 
 2   bmi       1338 non-null   float64
 3   children  1338 non-null   int64  
 4   smoker    1338 non-null   object 
 5   region    1338 non-null   object 
 6   charges   1338 non-null   float64
dtypes: float64(2), int64(2), object(3)
memory usage: 73.3+ KB
None
```

In [62]:

```python
# Age distribution visualization
plt.figure(figsize=(10, 6))
sns.histplot(insurance['age'], bins=20)
plt.title('Distribution of Age')
plt.xlabel('Age')
plt.show()
```



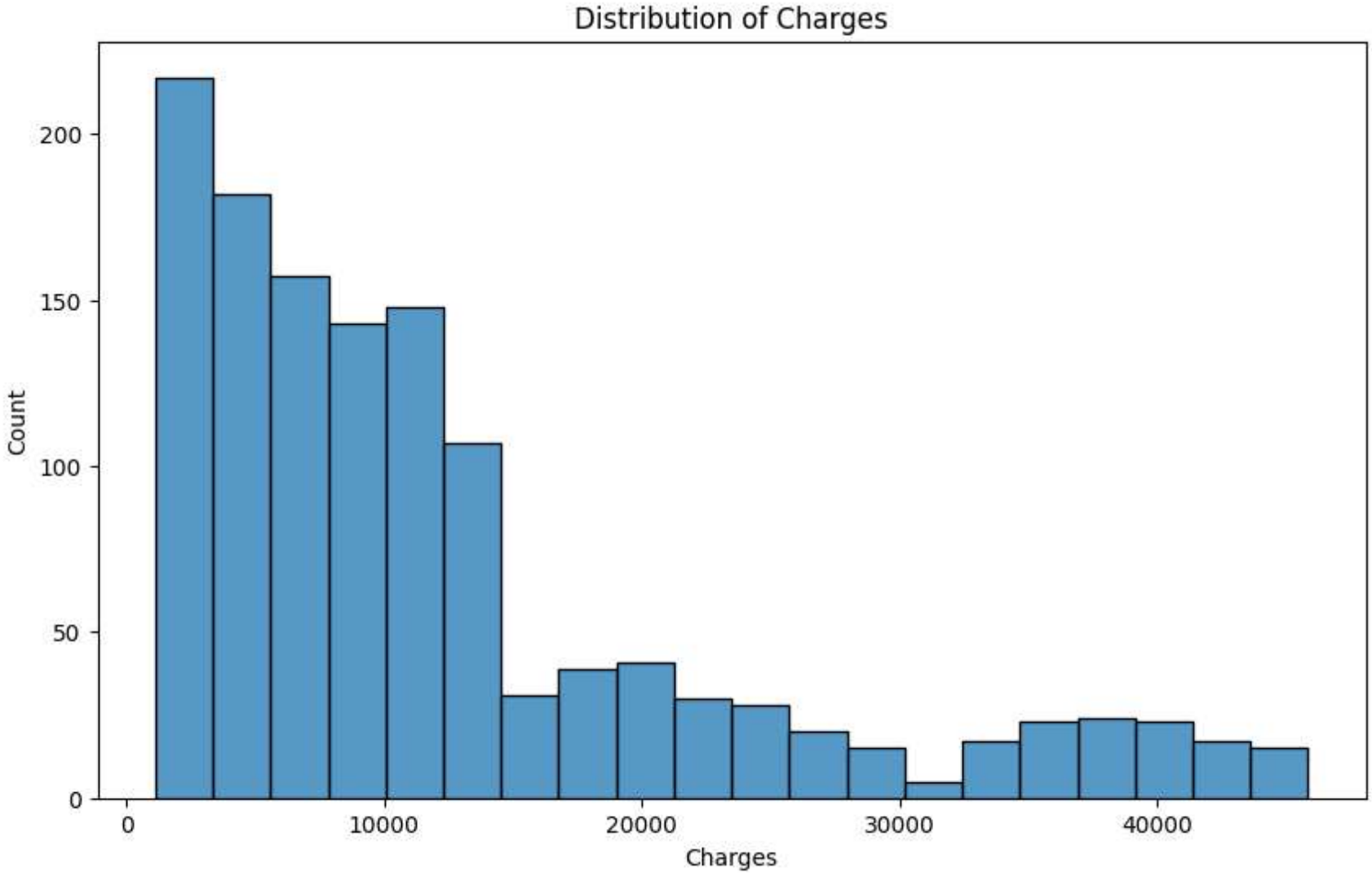
In [63]:
```
# Continuous and discrete features
numerical_continuous = ['age', 'bmi', 'charges']
numerical_discrete = ['children']
categorical = ['sex', 'smoker', 'region']
```

In [64]:
```
# Check for outliers in numerical_continuous columns
for col in numerical_continuous:
    z_scores = zscore(insurance[numerical_continuous])
```

```
    abs_z_scores = np.abs(z_scores)
    filtered_entries = (abs_z_scores < 3).all(axis=1)
    insurance = insurance[filtered_entries]
```

In [65]:

```
# Value distribution visualization
plt.figure(figsize=(10, 6))
sns.histplot(insurance['charges'], bins=20)
plt.title('Distribution of Charges')
plt.xlabel('Charges')
plt.show()
```



In [66]:
```
# Outlier detection and removal for discrete and categorical features
for col in numerical_discrete + categorical:
    threshold = 0.05
    counts = insurance[col].value_counts(normalize=True)
    mask = insurance[col].isin(counts[counts < threshold].index)
    insurance[col][mask] = 'Other'
```

In [68]:
```
insurance[numerical_discrete] = insurance[numerical_discrete].astype(str)
```

In [69]:
```
# Encoding categorical and discrete features
encoder = OneHotEncoder(sparse=False, handle_unknown='ignore')
X_encoded = encoder.fit_transform(insurance[categorical + numerical_discrete])
X_encoded = pd.DataFrame(X_encoded, columns=encoder.get_feature_names(categorical + numerical_discrete), index=insurance.index)
X = pd.concat([X_encoded, insurance[numerical_continuous]], axis=1)
y = insurance['charges']
```

```
C:\Users\EagleCORS\anaconda3\lib\site-packages\sklearn\utils\deprecation.py:87: FutureWarning: Function get_feature_names is dep
recated; get_feature_names is deprecated in 1.0 and will be removed in 1.2. Please use get_feature_names_out instead.
  warnings.warn(msg, category=FutureWarning)
```

In [71]:
```
# Feature selection
selector = SelectKBest(f_regression, k=5)
selector.fit(X, y)
X_new = selector.transform(X)
selected_features = X.columns[selector.get_support(indices=True)]
X = X[selected_features]
```

In [72]:
```
# Split the data into training and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

In [74]:
```
# Train the model
regressor = LinearRegression()
regressor.fit(X_train, y_train)
```

Out[74]:
```
LinearRegression()
```

In [75]:
```
# Evaluate the model
y_pred = regressor.predict(X_test)
print('MAE:', mean_absolute_error(y_test, y_pred))
```

```
MAE: 2.778028563781123e-12
```

In [76]:
```
print('MSE:', mean_squared_error(y_test, y_pred))
```

```
MSE: 2.327129425646509e-23
```

In [77]:
```
print('RMSE:', np.sqrt(mean_squared_error(y_test, y_pred)))
```

```
RMSE: 4.8240329866684255e-12
```

In [ ]: