BDA (MR22-1CS0234) Holiday Assignment

Question 1: Handling Imbalanced Datasets

In [2]:
```
# Import Required Libraries
from imblearn.over_sampling import SMOTE
from sklearn.datasets import make_classification
import pandas as pd

# Create synthetic imbalanced dataset
X, y = make_classification(n_classes=2, class_sep=2, weights=[0.9, 0.1],
                           n_informative=3, n_redundant=1, flip_y=0,
                           n_features=5, n_clusters_per_class=1, n_samples=1000, random_state=42)

# Before balancing
print("Original class distribution:", pd.Series(y).value_counts())

# Apply SMOTE
smote = SMOTE(random_state=42)
X_smote, y_smote = smote.fit_resample(X, y)

# After balancing
print("Balanced class distribution:", pd.Series(y_smote).value_counts())
```

```
Original class distribution: 0    900
1    100
Name: count, dtype: int64
Balanced class distribution: 0    900
1    900
Name: count, dtype: int64
```

Question 2: Optimal Clusters for K-means

In [3]:
```
# Import Libraries
from sklearn.cluster import KMeans
import matplotlib.pyplot as plt
from sklearn.datasets import make_blobs

# Create synthetic dataset for clustering
X_blobs, _ = make_blobs(n_samples=500, n_features=2, centers=4, random_state=42)

# Elbow Method
wcss = []
for k in range(1, 11):
    kmeans = KMeans(n_clusters=k, random_state=42)
    kmeans.fit(X_blobs)
    wcss.append(kmeans.inertia_)

# Plot Elbow Curve
plt.plot(range(1, 11), wcss, marker='o')
plt.title('Elbow Method')
plt.xlabel('Number of Clusters')
plt.ylabel('WCSS')
plt.show()
```

```
Unexpected exception formatting exception. Falling back to standard exception
Traceback (most recent call last):
  File "C:\Users\SHAIK TOUFIQSAHEB\AppData\Roaming\Python\Python312\site-packages\IPython\core\interactiveshell.
py", line 3577, in run_code
    exec(code_obj, self.user_global_ns, self.user_ns)
  File "C:\Users\SHAIK TOUFIQSAHEB\AppData\Local\Temp\ipykernel_17900\1171700513.py", line 13, in <module>
    kmeans.fit(X_blobs)
  File "C:\ProgramData\anaconda3\Lib\site-packages\sklearn\base.py", line 1474, in wrapper
    return fit_method(estimator, *args, **kwargs)
           ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "C:\ProgramData\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py", line 1519, in fit
    self._check_mkl_vcomp(X, X.shape[0])
  File "C:\ProgramData\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py", line 929, in _check_mkl_vcomp
    modules = threadpool_info()
              ^^^^^^^^^^^^^^^^^
  File "C:\ProgramData\anaconda3\Lib\site-packages\sklearn\utils\fixes.py", line 105, in threadpool_info
    return threadpoolctl.threadpool_info()
           ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "C:\ProgramData\anaconda3\Lib\site-packages\threadpoolctl.py", line 124, in threadpool_info
    return _ThreadpoolInfo(user_api=_ALL_USER_APIS).todicts()
           ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "C:\ProgramData\anaconda3\Lib\site-packages\threadpoolctl.py", line 340, in __init__
```

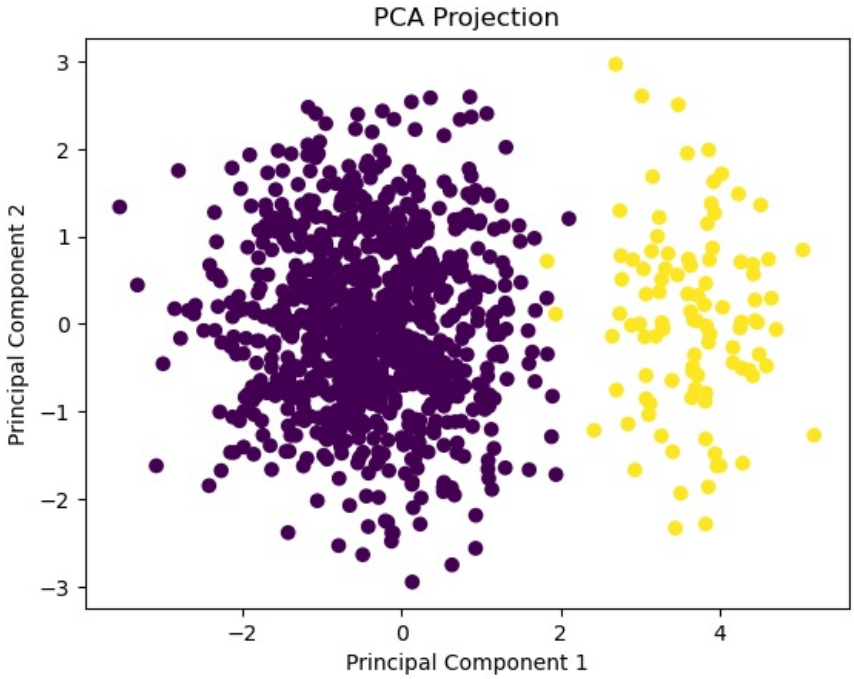
```
    self._load_modules()
  File "C:\ProgramData\anaconda3\Lib\site-packages\threadpoolctl.py", line 373, in _load_modules
    self._find_modules_with_enum_process_module_ex()
  File "C:\ProgramData\anaconda3\Lib\site-packages\threadpoolctl.py", line 485, in _find_modules_with_enum_proce
ss_module_ex
    self._make_module_from_path(filepath)
  File "C:\ProgramData\anaconda3\Lib\site-packages\threadpoolctl.py", line 515, in _make_module_from_path
    module = module_class(filepath, prefix, user_api, internal_api)
             ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "C:\ProgramData\anaconda3\Lib\site-packages\threadpoolctl.py", line 606, in __init__
    self.version = self.get_version()
                   ^^^^^^^^^^^^^^^^^^
  File "C:\ProgramData\anaconda3\Lib\site-packages\threadpoolctl.py", line 646, in get_version
    config = get_config().split()
             ^^^^^^^^^^^^^^^^^^
AttributeError: 'NoneType' object has no attribute 'split'

During handling of the above exception, another exception occurred:

Traceback (most recent call last):
  File "C:\Users\SHAIK TOUFIQSAHEB\AppData\Roaming\Python\Python312\site-packages\IPython\core\interactiveshell.
py", line 2168, in showtraceback
    stb = self.InteractiveTB.structured_traceback(
          ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "C:\Users\SHAIK TOUFIQSAHEB\AppData\Roaming\Python\Python312\site-packages\IPython\core\ultratb.py", line
1457, in structured_traceback
    return FormattedTB.structured_traceback(
           ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "C:\Users\SHAIK TOUFIQSAHEB\AppData\Roaming\Python\Python312\site-packages\IPython\core\ultratb.py", line
1348, in structured_traceback
    return VerboseTB.structured_traceback(
           ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "C:\Users\SHAIK TOUFIQSAHEB\AppData\Roaming\Python\Python312\site-packages\IPython\core\ultratb.py", line
1195, in structured_traceback
    formatted_exception = self.format_exception_as_a_whole(etype, evalue, etb, number_of_lines_of_context,
                          ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "C:\Users\SHAIK TOUFIQSAHEB\AppData\Roaming\Python\Python312\site-packages\IPython\core\ultratb.py", line
1110, in format_exception_as_a_whole
    frames.append(self.format_record(record))
                  ^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "C:\Users\SHAIK TOUFIQSAHEB\AppData\Roaming\Python\Python312\site-packages\IPython\core\ultratb.py", line
992, in format_record
    frame_info.lines, Colors, self.has_colors, lvals
    ^^^^^^^^^^^^^^^^
  File "C:\Users\SHAIK TOUFIQSAHEB\AppData\Roaming\Python\Python312\site-packages\IPython\core\ultratb.py", line
804, in lines
    return self._sd.lines
           ^^^^^^^^^^^^^^
  File "C:\Users\SHAIK TOUFIQSAHEB\AppData\Roaming\Python\Python312\site-packages\stack_data\utils.py", line 145
, in cached_property_wrapper
    value = obj.__dict__[self.func.__name__] = self.func(obj)
                                               ^^^^^^^^^^^^^^
  File "C:\Users\SHAIK TOUFIQSAHEB\AppData\Roaming\Python\Python312\site-packages\stack_data\core.py", line 734,
in lines
    pieces = self.included_pieces
             ^^^^^^^^^^^^^^^^^^^^
  File "C:\Users\SHAIK TOUFIQSAHEB\AppData\Roaming\Python\Python312\site-packages\stack_data\utils.py", line 145
, in cached_property_wrapper
    value = obj.__dict__[self.func.__name__] = self.func(obj)
                                               ^^^^^^^^^^^^^^
  File "C:\Users\SHAIK TOUFIQSAHEB\AppData\Roaming\Python\Python312\site-packages\stack_data\core.py", line 677,
in included_pieces
    scope_pieces = self.scope_pieces
                   ^^^^^^^^^^^^^^^^^
  File "C:\Users\SHAIK TOUFIQSAHEB\AppData\Roaming\Python\Python312\site-packages\stack_data\utils.py", line 145
, in cached_property_wrapper
    value = obj.__dict__[self.func.__name__] = self.func(obj)
                                               ^^^^^^^^^^^^^^
  File "C:\Users\SHAIK TOUFIQSAHEB\AppData\Roaming\Python\Python312\site-packages\stack_data\core.py", line 614,
in scope_pieces
    scope_start, scope_end = self.source.line_range(self.scope)
                             ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "C:\Users\SHAIK TOUFIQSAHEB\AppData\Roaming\Python\Python312\site-packages\stack_data\core.py", line 178,
in line_range
    return line_range(self.asttext(), node)
                      ^^^^^^^^^^^^
AttributeError: 'Source' object has no attribute 'asttext'
```

Question 3: Dimensionality Reduction (PCA)

In [4]:
```
# Import Libraries
from sklearn.decomposition import PCA
```

```
import numpy as np

# Apply PCA
pca = PCA(n_components=2)
X_pca = pca.fit_transform(X)

print("Explained Variance Ratio:", pca.explained_variance_ratio_)

# Scatter plot for PCA projection
plt.scatter(X_pca[:, 0], X_pca[:, 1], c=y, cmap='viridis')
plt.title('PCA Projection')
plt.xlabel('Principal Component 1')
plt.ylabel('Principal Component 2')
plt.show()
```

```
Explained Variance Ratio: [0.52205186 0.22886396]
```

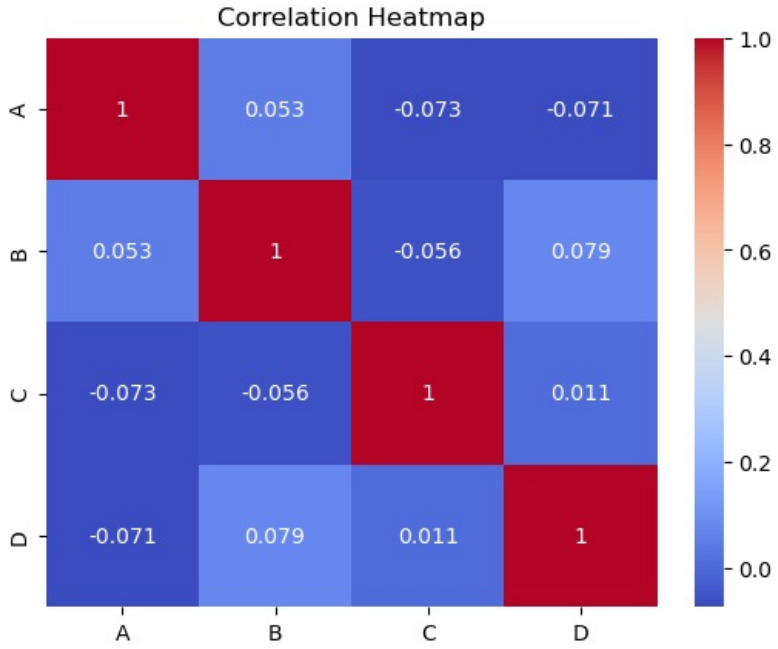


Question 4: Correlations in a Dataset

In [5]:

```
# Create a synthetic dataset
import seaborn as sns
import numpy as np
import pandas as pd

data = pd.DataFrame({
    'A': np.random.rand(100),
    'B': np.random.rand(100),
    'C': np.random.rand(100),
    'D': np.random.rand(100)
})

# Compute correlations
correlation_matrix = data.corr()
print(correlation_matrix)

# Heatmap
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm')
plt.title('Correlation Heatmap')
plt.show()
```

```
          A         B         C         D
A  1.000000  0.052511 -0.073208 -0.071368
B  0.052511  1.000000 -0.056459  0.078948
C -0.073208 -0.056459  1.000000  0.010580
D -0.071368  0.078948  0.010580  1.000000
```



Question 5: Handling Missing Values

In [6]:
```
from sklearn.impute import SimpleImputer

# Create data with missing values
data_with_nans = pd.DataFrame({
    'A': [1, 2, np.nan, 4],
    'B': [np.nan, 2, 3, 4],
    'C': [1, np.nan, np.nan, 4]
})

print("Original Data with NaNs:")
print(data_with_nans)

# Imputation
imputer = SimpleImputer(strategy='mean')
data_imputed = pd.DataFrame(imputer.fit_transform(data_with_nans), columns=data_with_nans.columns)

print("Data after Imputation:")
print(data_imputed)
```

```
Original Data with NaNs:
     A    B    C
0  1.0  NaN  1.0
1  2.0  2.0  NaN
2  NaN  3.0  NaN
3  4.0  4.0  4.0
Data after Imputation:
          A    B    C
0  1.000000  3.0  1.0
1  2.000000  2.0  2.5
2  2.333333  3.0  2.5
3  4.000000  4.0  4.0
```

Question 6: Detect and Remove Duplicates

In [7]:
```
# Create data with duplicates
data_with_duplicates = pd.DataFrame({
    'A': [1, 2, 2, 4],
    'B': [5, 6, 6, 8],
    'C': [9, 10, 10, 12]
})
```

```
print("Original Data:")
print(data_with_duplicates)

# Remove duplicates
data_no_duplicates = data_with_duplicates.drop_duplicates()
print("Data after removing duplicates:")
print(data_no_duplicates)
```

```
Original Data:
   A  B   C
0  1  5   9
1  2  6  10
2  2  6  10
3  4  8  12
Data after removing duplicates:
   A  B   C
0  1  5   9
1  2  6  10
3  4  8  12
```

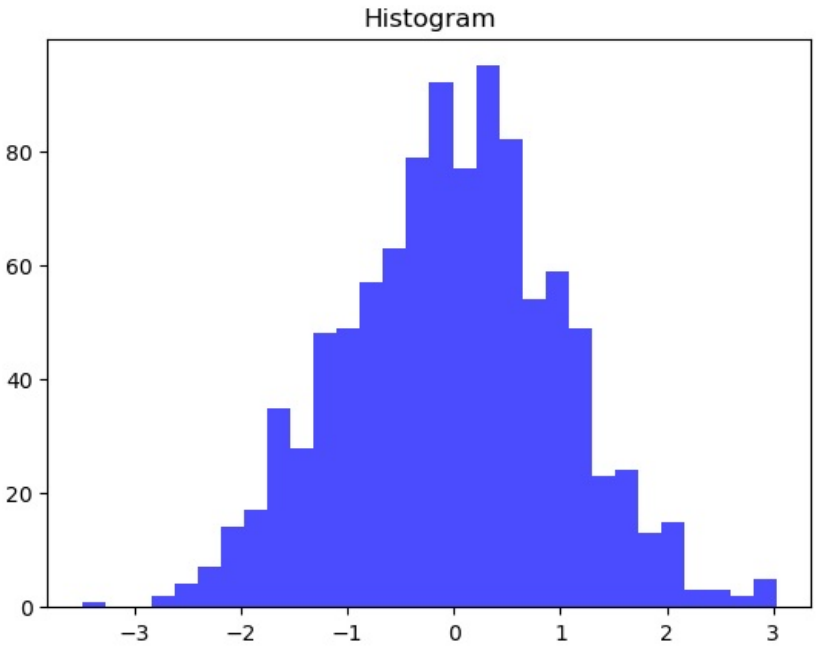
Question 7: Random Forest Regression for Housing Prices

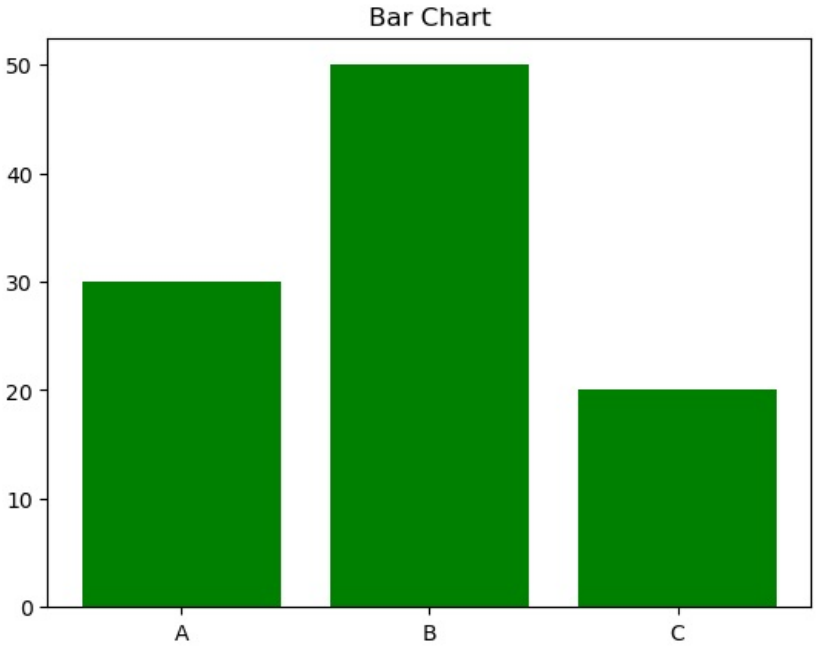
In [8]:
```
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error
from sklearn.model_selection import train_test_split

# Create synthetic housing dataset
X_housing, y_housing = make_classification(n_samples=1000, n_features=5, random_state=42)
X_train, X_test, y_train, y_test = train_test_split(X_housing, y_housing, test_size=0.3, random_state=42)

# Train Random Forest Regressor
rf = RandomForestRegressor(n_estimators=100, random_state=42)
rf.fit(X_train, y_train)

# Predictions
y_pred = rf.predict(X_test)
print("MSE:", mean_squared_error(y_test, y_pred))
```

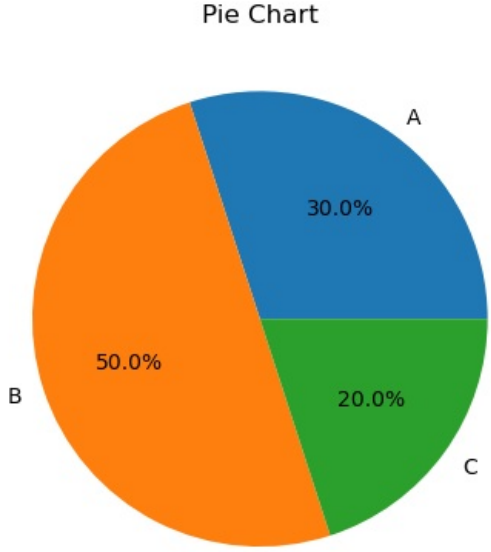
```
MSE: 0.06833633333333333
```

Question 8: Histogram, Bar Chart, and Pie Chart

In [9]:
```
# Create sample dataset
data_chart = pd.DataFrame({
    'Category': ['A', 'B', 'C'],
    'Values': [30, 50, 20]
})

# Histogram
plt.hist(np.random.randn(1000), bins=30, color='blue', alpha=0.7)
plt.title('Histogram')
plt.show()

# Bar Chart
plt.bar(data_chart['Category'], data_chart['Values'], color='green')
plt.title('Bar Chart')
plt.show()

# Pie Chart
plt.pie(data_chart['Values'], labels=data_chart['Category'], autopct='%1.1f%%')
plt.title('Pie Chart')
plt.show()
```







Question 9: Linear and Logistic Regression

In [10]:
```
from sklearn.linear_model import LinearRegression, LogisticRegression
from sklearn.metrics import accuracy_score
```

```
# Linear Regression
linear_model = LinearRegression()
linear_model.fit(X_train, y_train)

# Predictions and evaluation for Linear Regression
y_pred_linear = linear_model.predict(X_test)
print("Linear Regression MSE:", mean_squared_error(y_test, y_pred_linear))

# Logistic Regression
logistic_model = LogisticRegression()
logistic_model.fit(X_train, y_train)

# Predictions and evaluation for Logistic Regression
y_pred_logistic = logistic_model.predict(X_test)
print("Logistic Regression Accuracy:", accuracy_score(y_test, y_pred_logistic))
```

```
Linear Regression MSE: 0.11065706186080641
Logistic Regression Accuracy: 0.89
```

Question 10: Lag Features for Time-Series Data

In [11]:
```
# Create time-series dataset

time_series_data = pd.DataFrame({'Date': pd.date_range(start='1/1/2020', periods=10, freq='D'),
                                 'Value': np.random.rand(10)})
time_series_data['Lag_1'] = time_series_data['Value'].shift(1)
time_series_data['Lag_2'] = time_series_data['Value'].shift(2)

print("Time-Series Data with Lag Features:")
print(time_series_data)
```

```
Time-Series Data with Lag Features:
        Date     Value     Lag_1     Lag_2
0 2020-01-01  0.617709       NaN       NaN
1 2020-01-02  0.537258  0.617709       NaN
2 2020-01-03  0.988252  0.537258  0.617709
3 2020-01-04  0.384586  0.988252  0.537258
4 2020-01-05  0.169331  0.384586  0.988252
5 2020-01-06  0.634545  0.169331  0.384586
6 2020-01-07  0.503799  0.634545  0.169331
7 2020-01-08  0.245256  0.503799  0.634545
8 2020-01-09  0.738379  0.245256  0.503799
9 2020-01-10  0.316331  0.738379  0.245256
```

In [ ]:
```

```