MSA 2025 Phase 2 - Part 1

```python
import sklearn
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
```

1. Find all variables and understand them

```
features = pd.read_csv('datasets/W store sales/features.csv')
stores = pd.read_csv('datasets/W store sales/stores.csv')
sales = pd.read_csv('datasets/W store sales/sales.csv')

# Display first 10 rows
print('First 10 rows of features.csv:')
display(features.head(10))
print('First 10 rows of stores.csv:')
display(stores.head(10))
print('First 10 rows of sales.csv:')
display(sales.head(10))

# Statistical summary
print('Statistical summary of features.csv:')
display(features.describe())
print('Statistical summary of stores.csv:')
display(stores.describe())
print('Statistical summary of sales.csv:')
display(sales.describe())

# Data types
print('Data types in features.csv:')
print(features.dtypes)
print('Data types in stores.csv:')
print(stores.dtypes)
print('Data types in sales.csv:')
print(sales.dtypes)

# Merge store info into features
df = features.merge(stores, on='Store', how='left').merge(sales, on=['Store', 'Date'],
how='left')

print(f"Number of instances: {df.shape[0]}, Number of features: {df.shape[1]}")

# First check which columns are in the merged DataFrame
print("Columns in merged dataframe:")
print(df.columns.tolist())

# Handle duplicate IsHoliday columns from merge
if 'IsHoliday_x' in df.columns:
    df['IsHoliday'] = df['IsHoliday_x'].astype(int)  # Use features.csv version
    df.drop(['IsHoliday_x', 'IsHoliday_y'], axis=1, inplace=True)  # Remove duplicates
    print("IsHoliday column converted to numeric")
elif 'IsHoliday' in df.columns:
    df['IsHoliday'] = df['IsHoliday'].astype(int)
    print("IsHoliday column converted to numeric")
else:
    print("IsHoliday column not found in dataframe")
```

```python
if 'Type' in df.columns:
    df['Type'] = df['Type'].map({'A': 0, 'B': 1, 'C': 2})
    print("Type column converted to numeric")
else:
    print("Type column not found in dataframe")
```

```
First 10 rows of features.csv:
```

	Store	Date	Temperature	Fuel_Price	MarkDown1	MarkDown2	MarkDown3	MarkDown4
0	1	2010-02-05	42.31	2.572	NaN	NaN	NaN	NaN
1	1	2010-02-12	38.51	2.548	NaN	NaN	NaN	NaN
2	1	2010-02-19	39.93	2.514	NaN	NaN	NaN	NaN
3	1	2010-02-26	46.63	2.561	NaN	NaN	NaN	NaN
4	1	2010-03-05	46.50	2.625	NaN	NaN	NaN	NaN
5	1	2010-03-12	57.79	2.667	NaN	NaN	NaN	NaN
6	1	2010-03-19	54.58	2.720	NaN	NaN	NaN	NaN
7	1	2010-03-26	51.45	2.732	NaN	NaN	NaN	NaN
8	1	2010-04-02	62.27	2.719	NaN	NaN	NaN	NaN
9	1	2010-04-09	65.86	2.770	NaN	NaN	NaN	NaN

```
First 10 rows of stores.csv:
```

	Store	Type	Size
0	1	A	151315
1	2	A	202307
2	3	B	37392
3	4	A	205863
4	5	B	34875
5	6	A	202505
6	7	B	70713
7	8	A	155078
8	9	B	125833
9	10	B	126512

```
First 10 rows of sales.csv:
```

	Store	Dept	Date	Weekly_Sales	IsHoliday
0	1	1	2010-02-05	24924.50	False
1	1	1	2010-02-12	46039.49	True
2	1	1	2010-02-19	41595.55	False
3	1	1	2010-02-26	19403.54	False
4	1	1	2010-03-05	21827.90	False
5	1	1	2010-03-12	21043.39	False
6	1	1	2010-03-19	22136.64	False
7	1	1	2010-03-26	26229.21	False
8	1	1	2010-04-02	57258.43	False
9	1	1	2010-04-09	42960.91	False

```
Statistical summary of features.csv:
```

	Store	Temperature	Fuel_Price	MarkDown1	MarkDown2	MarkDown3
count	8190.000000	8190.000000	8190.000000	4032.000000	2921.000000	3613.000000
mean	23.000000	59.356198	3.405992	7032.371786	3384.176594	1760.100180
std	12.987966	18.678607	0.431337	9262.747448	8793.583016	11276.462208
min	1.000000	-7.290000	2.472000	-2781.450000	-265.760000	-179.260000
25%	12.000000	45.902500	3.041000	1577.532500	68.880000	6.600000
50%	23.000000	60.710000	3.513000	4743.580000	364.570000	36.260000
75%	34.000000	73.880000	3.743000	8923.310000	2153.350000	163.150000
max	45.000000	101.950000	4.468000	103184.980000	104519.540000	149483.310000

```
Statistical summary of stores.csv:
```

	Store	Size
count	45.000000	45.000000
mean	23.000000	130287.600000
std	13.133926	63825.271991
min	1.000000	34875.000000
25%	12.000000	70713.000000
50%	23.000000	126512.000000
75%	34.000000	202307.000000
max	45.000000	219622.000000

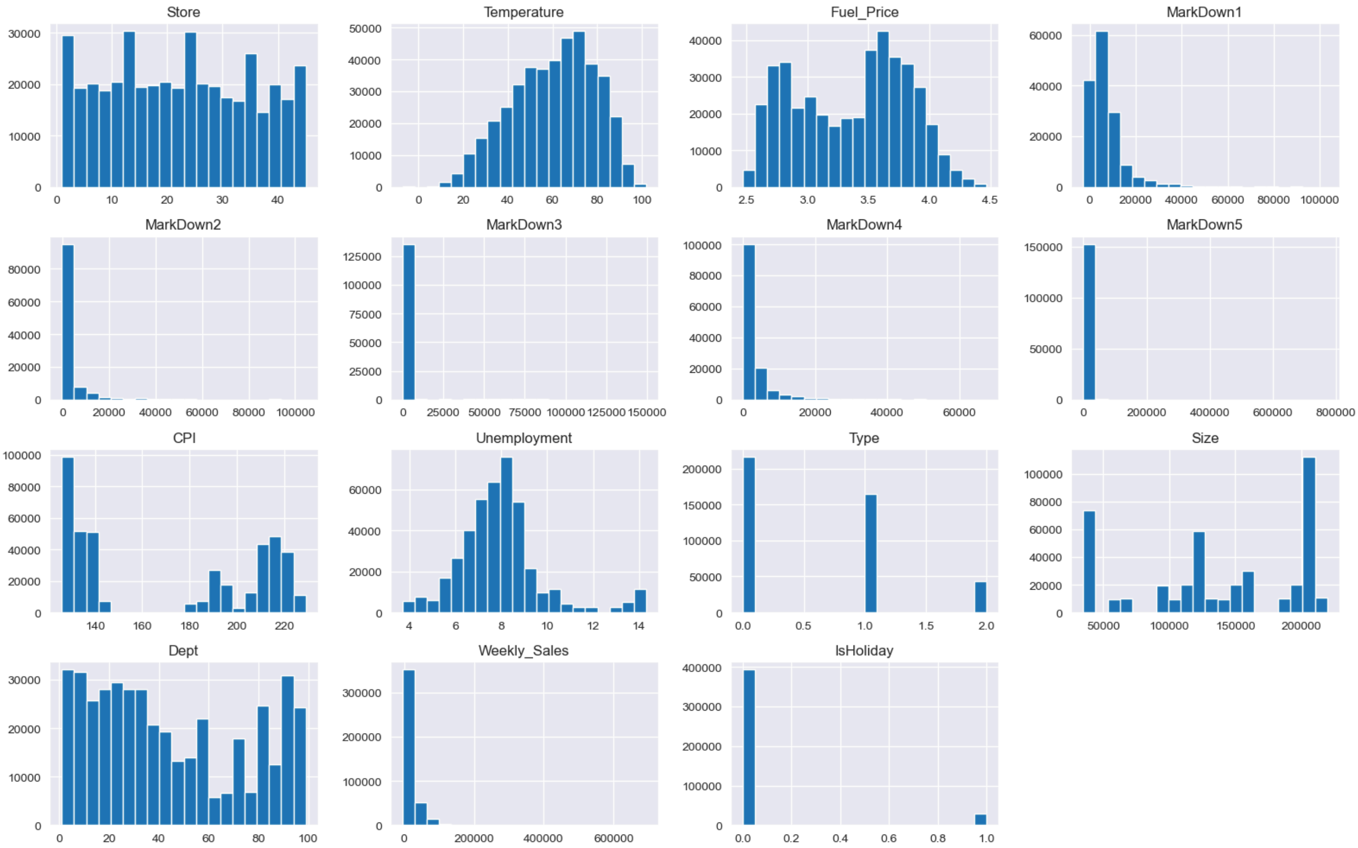
```
Statistical summary of sales.csv:
```

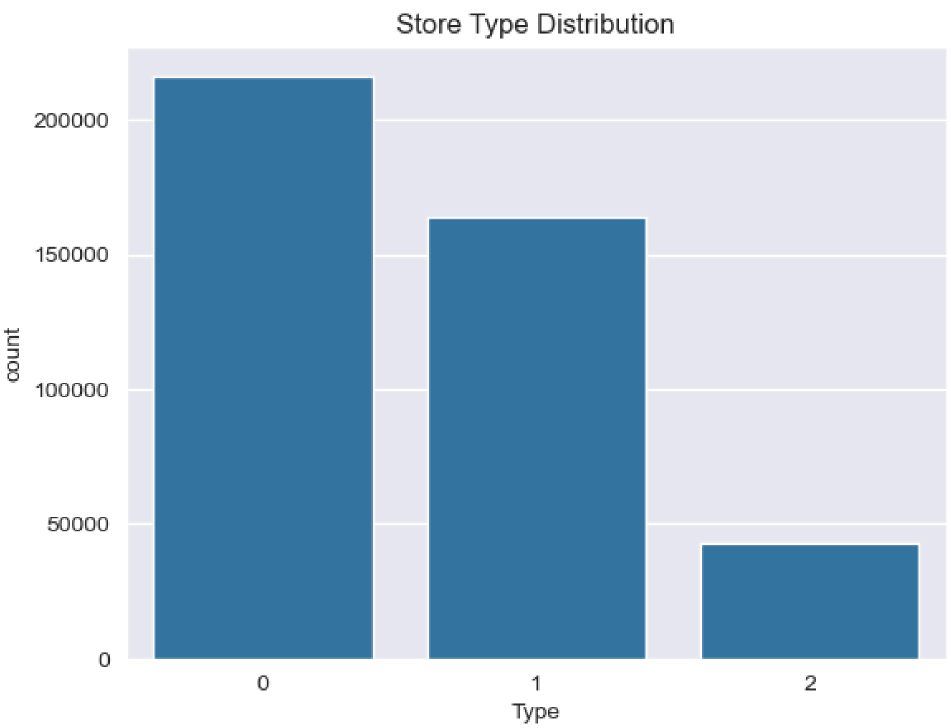
	Store	Dept	Weekly_Sales
count	421570.000000	421570.000000	421570.000000
mean	22.200546	44.260317	15981.258123
std	12.785297	30.492054	22711.183519
min	1.000000	1.000000	-4988.940000
25%	11.000000	18.000000	2079.650000
50%	22.000000	37.000000	7612.030000
75%	33.000000	74.000000	20205.852500
max	45.000000	99.000000	693099.360000

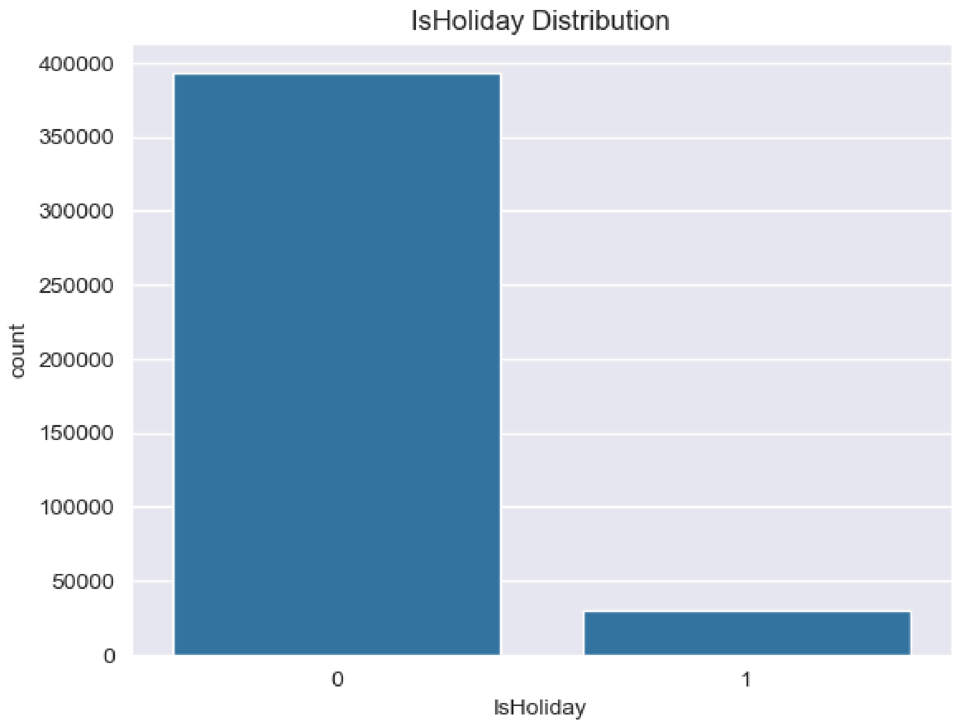
```
Data types in features.csv:
Store             int64
Date             object
Temperature     float64
Fuel_Price      float64
MarkDown1       float64
MarkDown2       float64
MarkDown3       float64
MarkDown4       float64
MarkDown5       float64
CPI             float64
Unemployment    float64
IsHoliday          bool
dtype: object
Data types in stores.csv:
Store     int64
Type     object
Size      int64
dtype: object
Data types in sales.csv:
Store             int64
Dept              int64
Date             object
Weekly_Sales    float64
IsHoliday          bool
dtype: object
Number of instances: 423325, Number of features: 17
Columns in merged dataframe:
['Store', 'Date', 'Temperature', 'Fuel_Price', 'MarkDown1', 'MarkDown2', 'MarkDown3', 'MarkDown4',
'MarkDown5', 'CPI', 'Unemployment', 'IsHoliday_x', 'Type', 'Size', 'Dept', 'Weekly_Sales', 'IsHoliday_y']
IsHoliday column converted to numeric
Type column converted to numeric
```

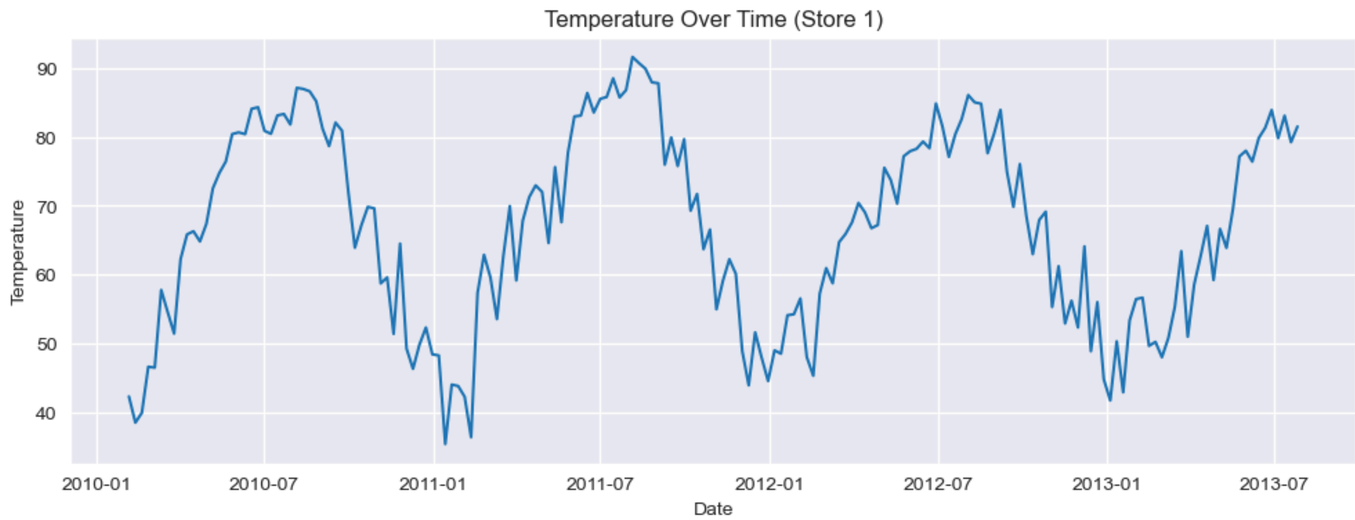
2. Visualise data

```
# Visualize numeric columns
df.hist(bins=20, figsize=(16, 10))
plt.tight_layout()
plt.show()

# Store type distribution
sns.countplot(x='Type', data=df)
plt.title('Store Type Distribution')
plt.show()

# Holiday distribution
sns.countplot(x='IsHoliday', data=df)
plt.title('IsHoliday Distribution')
plt.show()

# Time series visualization: Temperature over time for one store
sample_store = df.loc[df['Store'] == 1].copy()
sample_store['Date'] = pd.to_datetime(sample_store['Date'])
sample_store = sample_store.sort_values('Date')
plt.figure(figsize=(12, 4))
plt.plot(sample_store['Date'], sample_store['Temperature'])
plt.title('Temperature Over Time (Store 1)')
plt.xlabel('Date')
plt.ylabel('Temperature')
plt.show()

# Scatter plot of key features
pd.plotting.scatter_matrix(df[['Temperature', 'Fuel_Price', 'CPI', 'Unemployment']],
                           figsize=(12, 8), alpha=0.6)
plt.suptitle('Scatter Matrix of Key Features')
plt.show()
```

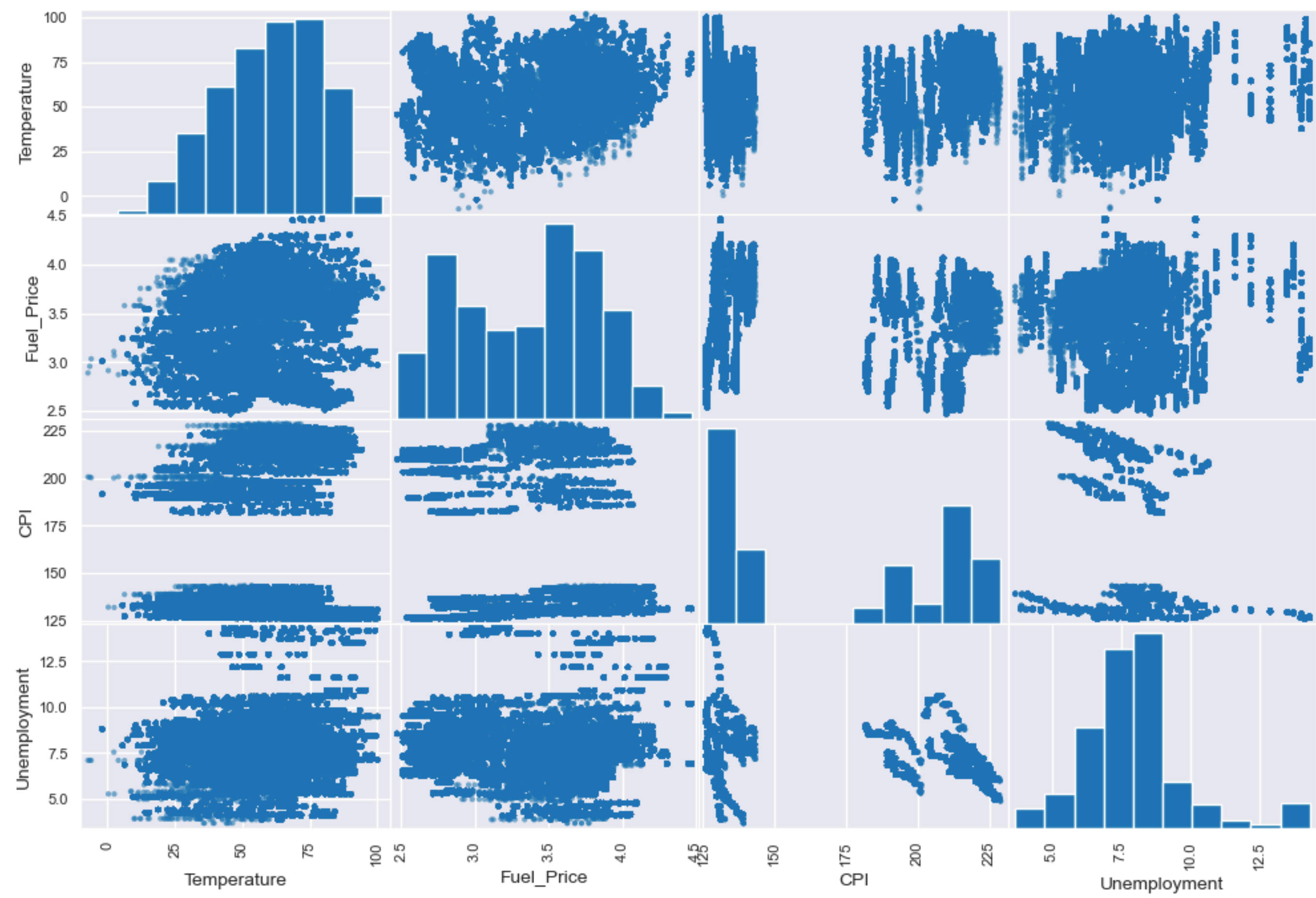

Store
Temperature
Fuel_Price
MarkDown1
MarkDown2
MarkDown3
MarkDown4
MarkDown5
CPI
Unemployment
Type
Size
Dept
Weekly_Sales
IsHoliday


Store Type Distribution
count
Type
0
1
2
200000
150000
100000
50000
0


IsHoliday Distribution
400000
350000
300000
250000
200000
150000
100000
50000
0
count
0
1
IsHoliday


Temperature Over Time (Store 1)
90
80
70
60
50
40
Temperature
2010-01
2010-07
2011-01
2011-07
2012-01
2012-07
2013-01
2013-07
Date

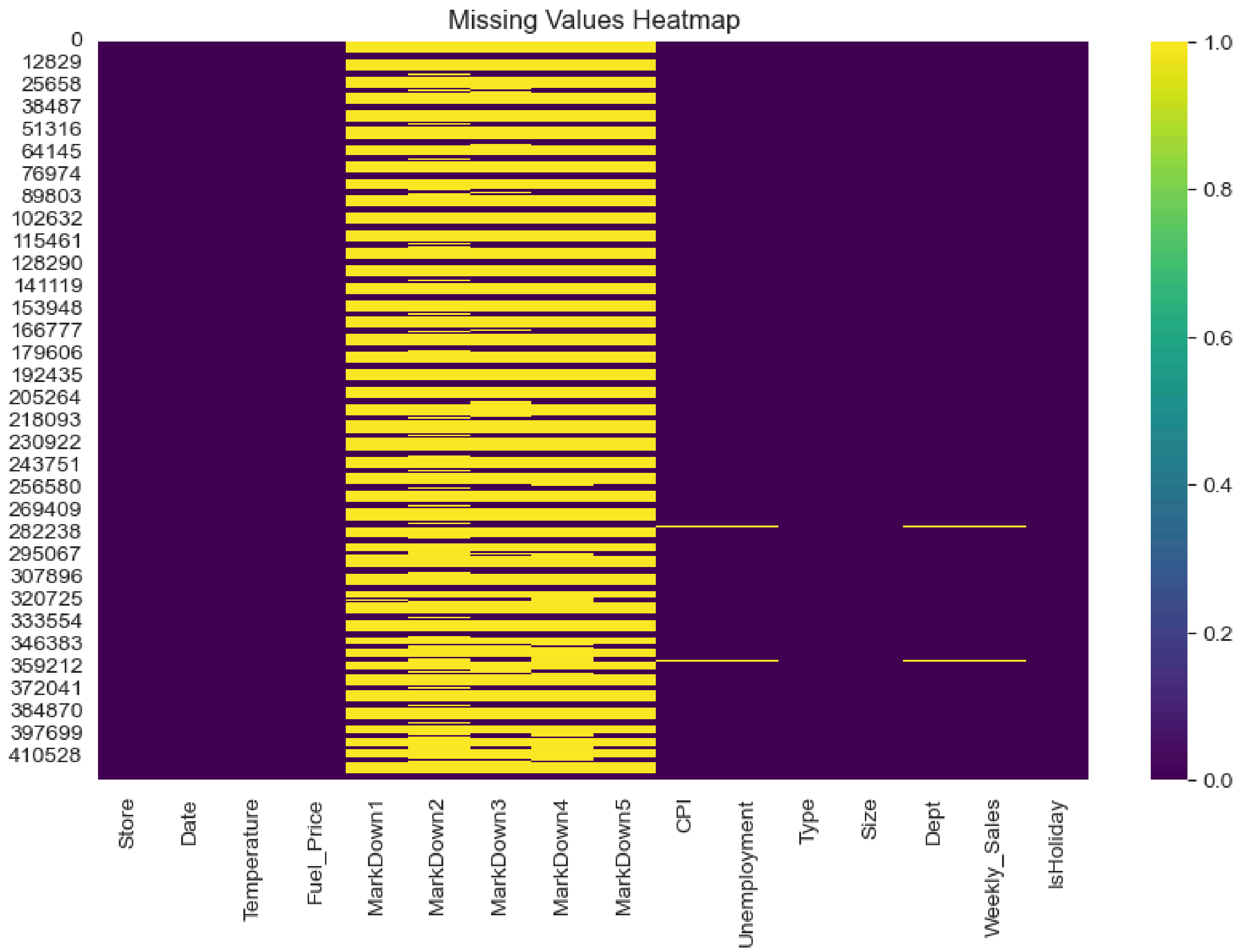
Scatter Matrix of Key Features



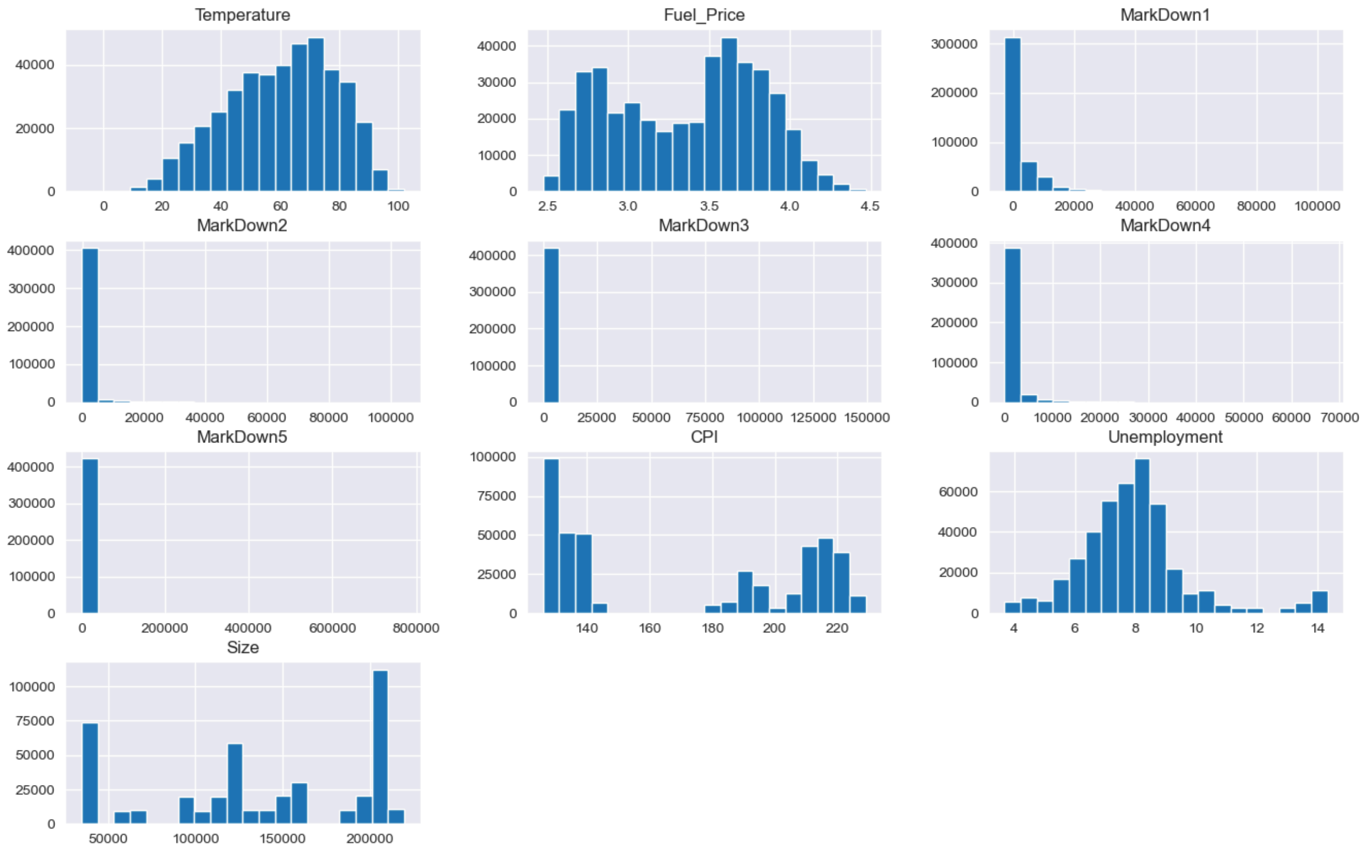
3. Clean data

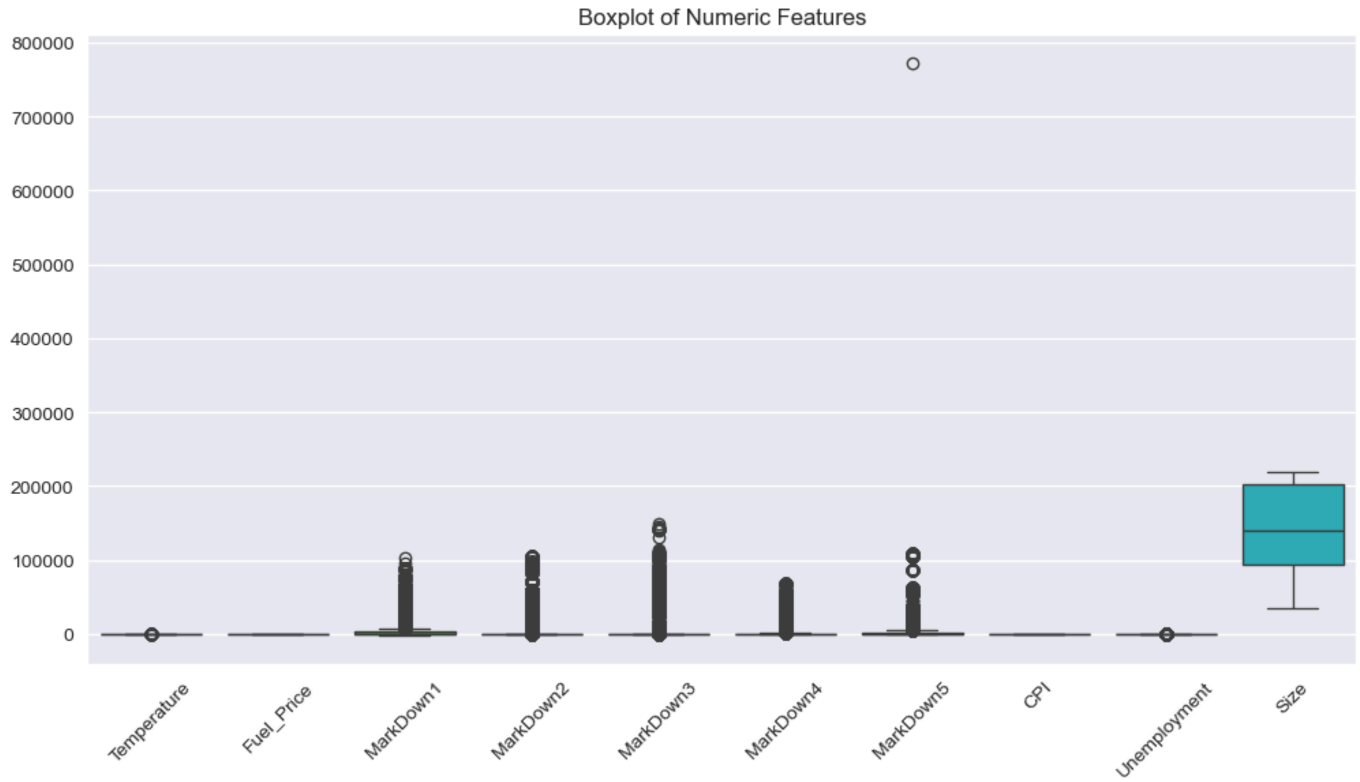
```
# Check missing values (show as table)
missing_counts = df.isnull().sum()
print('Missing values count:')
display(missing_counts[missing_counts > 0])

# Visualize missing values as a heatmap
plt.figure(figsize=(10, 6))
sns.heatmap(df.isnull(), cbar=True, cmap='viridis')
plt.title('Missing Values Heatmap')
plt.show()

# Fill MarkDown NAs with 0
for col in ['MarkDown1', 'MarkDown2', 'MarkDown3', 'MarkDown4', 'MarkDown5']:
    df[col] = df[col].fillna(0)
# Check again
missing_counts_after = df.isnull().sum()
print('\n\nMissing values after filling:')
display(missing_counts_after[missing_counts_after > 0])

# Outlier detection (boxplot)
num_cols = ['Temperature', 'Fuel_Price', 'MarkDown1', 'MarkDown2', 'MarkDown3',
'MarkDown4', 'MarkDown5', 'CPI',
            'Unemployment', 'Size']
df[num_cols].hist(bins=20, figsize=(16, 10))
plt.figure(figsize=(12, 6))
sns.boxplot(data=df[num_cols])
plt.title('Boxplot of Numeric Features')
plt.xticks(rotation=45)
plt.show()

scaler = sklearn.preprocessing.StandardScaler()
df[num_cols] = scaler.fit_transform(df[num_cols])
```

```
Missing values count:



MarkDown1       270892
MarkDown2       310793
MarkDown3       284667
MarkDown4       286859
MarkDown5       270138
CPI                585
Unemployment       585
Dept              1755
Weekly_Sales      1755
dtype: int64
```



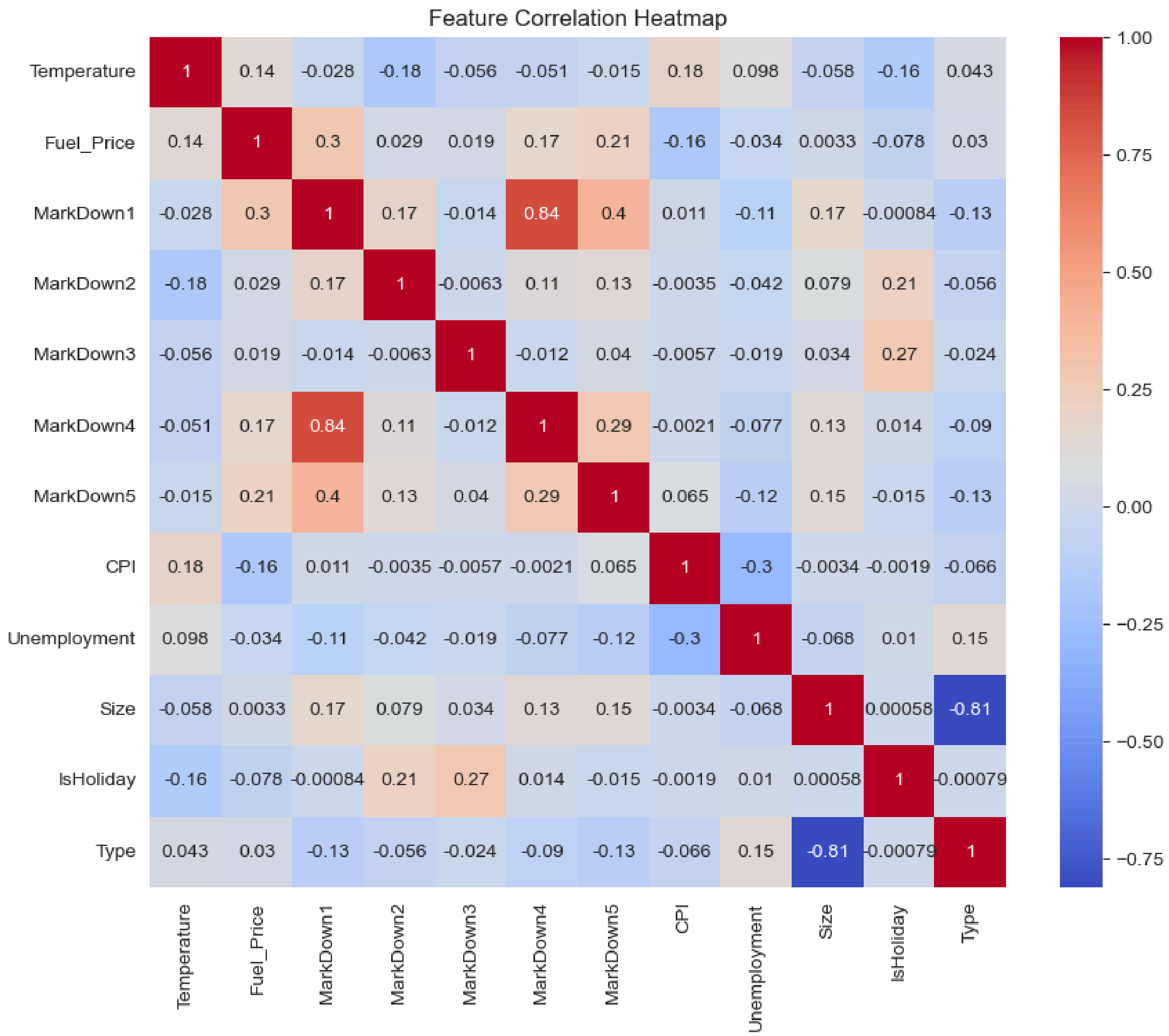

```
Missing values after filling:



CPI               585
Unemployment      585
Dept             1755
Weekly_Sales     1755
dtype: int64
```


Temperature
Fuel_Price
MarkDown1
MarkDown2
MarkDown3
MarkDown4
MarkDown5
CPI
Unemployment
Size


Boxplot of Numeric Features
Temperature
Fuel_Price
MarkDown1
MarkDown2
MarkDown3
MarkDown4
MarkDown5
CPI
Unemployment
Size

4. Identify correlated variables

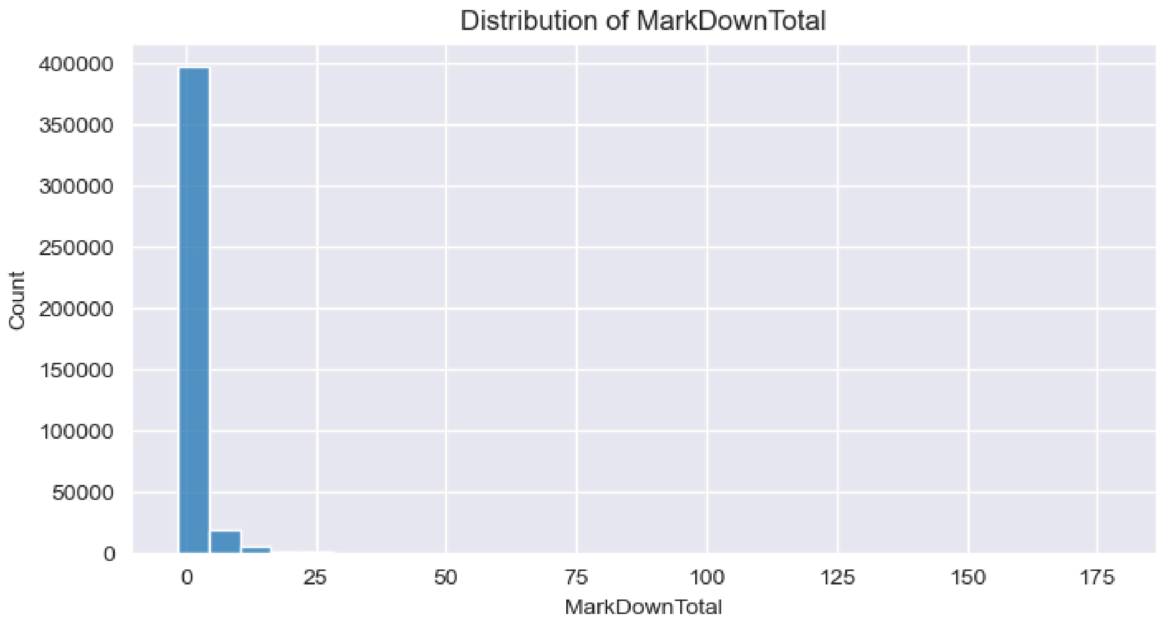
```python
# Correlation Analysis
corr = df[num_cols + ['IsHoliday', 'Type']].corr()
plt.figure(figsize=(10, 8))
sns.heatmap(corr, annot=True, cmap='coolwarm')
plt.title('Feature Correlation Heatmap')
plt.show()
```

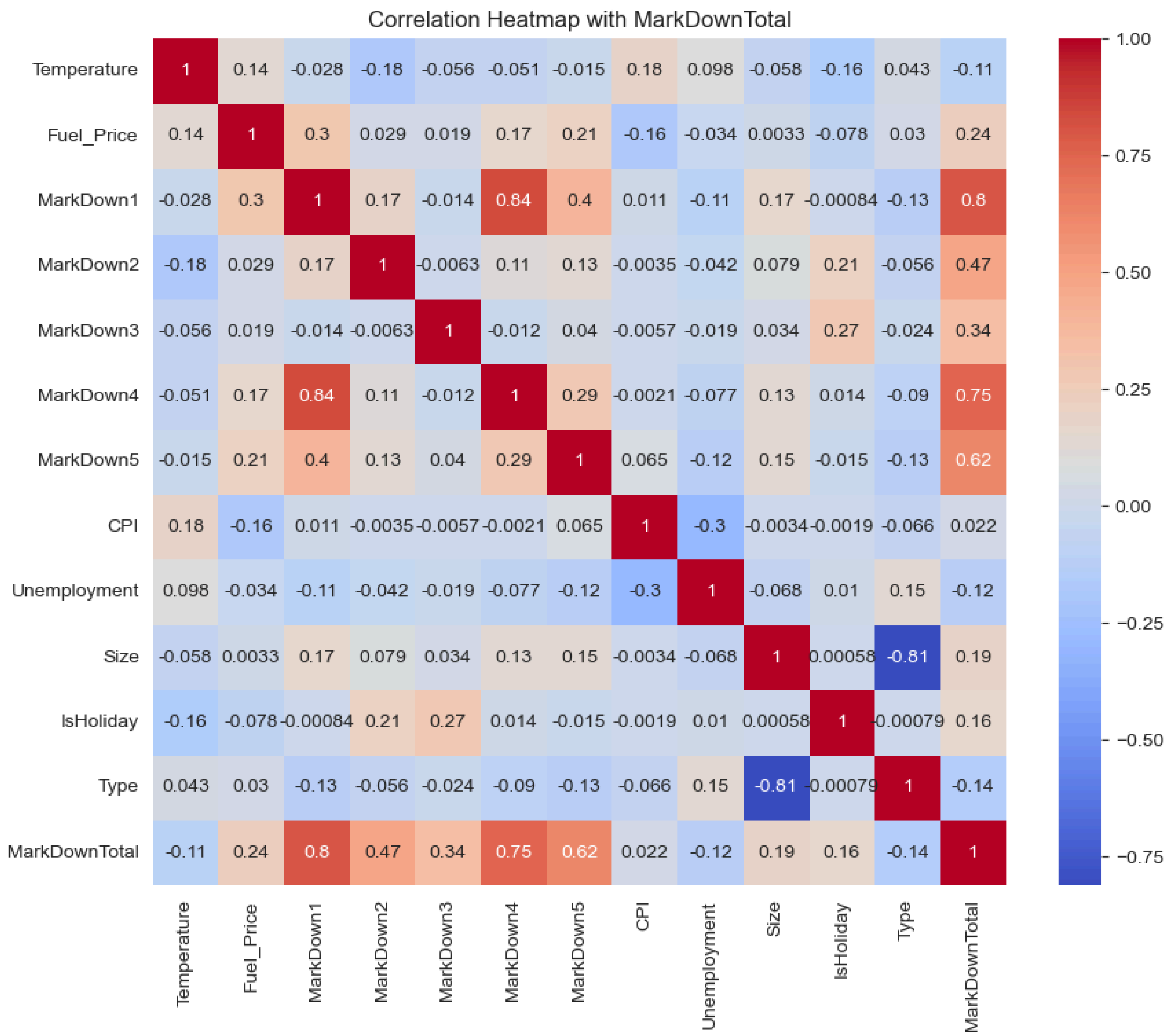


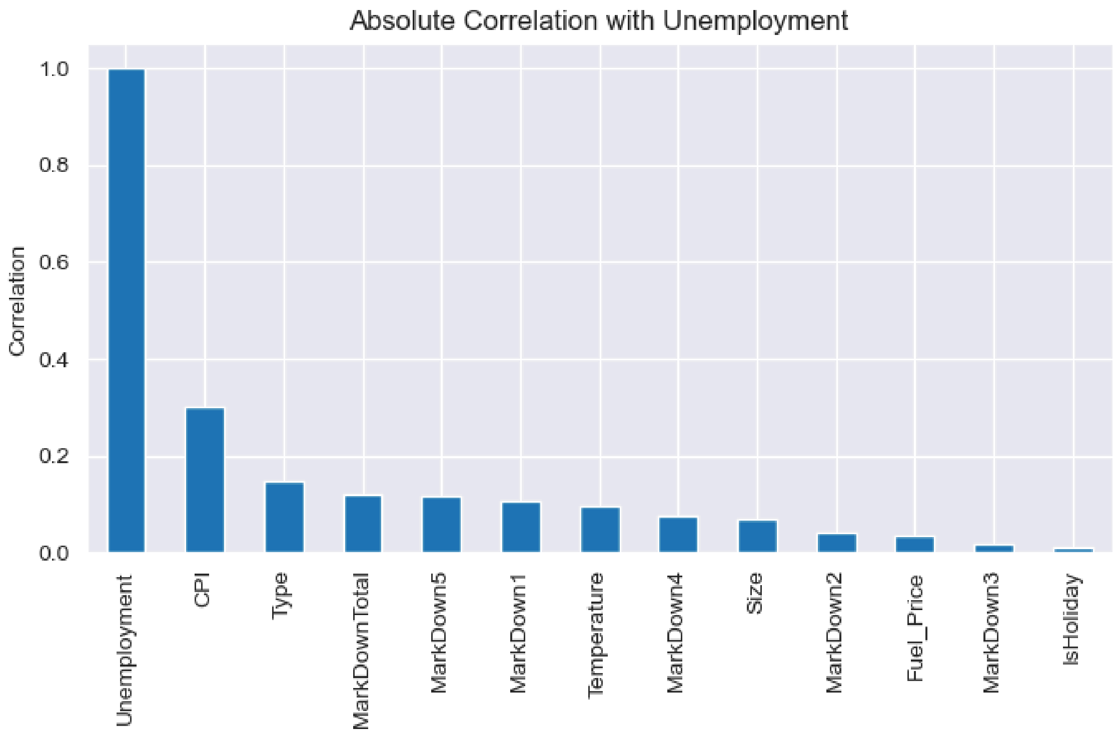
5. Feature Selection and Engineering (Optional Task)

```
# Feature correlation with Unemployment (as an example)
cor_target = abs(corr['Unemployment'])
relevant_features = cor_target[cor_target > 0.2].index.tolist()
print('Features highly correlated with Unemployment:', relevant_features)

# Feature engineering: MarkDownTotal
if not 'MarkDownTotal' in df.columns:
    df['MarkDownTotal'] = df[['MarkDown1', 'MarkDown2', 'MarkDown3', 'MarkDown4',
'MarkDown5']].sum(axis=1)

# Visualize new feature
plt.figure(figsize=(8, 4))
sns.histplot(df['MarkDownTotal'], bins=30)
plt.title('Distribution of MarkDownTotal')
plt.xlabel('MarkDownTotal')
plt.show()

# Correlation of new feature
corr2 = df[num_cols + ['IsHoliday', 'Type', 'MarkDownTotal']].corr()
plt.figure(figsize=(10, 8))
sns.heatmap(corr2, annot=True, cmap='coolwarm')
plt.title('Correlation Heatmap with MarkDownTotal')
plt.show()

# Feature importance (simple): visualize absolute correlation with Unemployment
abs_corr = corr2['Unemployment'].abs().sort_values(ascending=False)
plt.figure(figsize=(8, 4))
abs_corr.plot(kind='bar')
plt.title('Absolute Correlation with Unemployment')
plt.ylabel('Correlation')
plt.show()
```

```
Features highly correlated with Unemployment: ['CPI', 'Unemployment']
```


Distribution of MarkDownTotal
Count
MarkDownTotal


Correlation Heatmap with MarkDownTotal
Temperature
Fuel_Price
MarkDown1
MarkDown2
MarkDown3
MarkDown4
MarkDown5
CPI
Unemployment
Size
IsHoliday
Type
MarkDownTotal



6. Summary

This comprehensive analysis of the W store sales dataset demonstrates a systematic approach to exploratory data analysis and data preprocessing, preparing the data for future machine learning applications. The dataset combines three key components: features, stores, and sales data, resulting in a rich dataset with 423,325 instances across 17 features after merging.

The initial data exploration revealed the complexity of retail sales data, with three interconnected datasets requiring careful merging to preserve data integrity. The merged dataset contains temporal features such as temperature and fuel prices, economic indicators including CPI and unemployment rates, store characteristics like type and size, and promotional activities through markdown columns. A key challenge addressed was handling duplicate column names during the merge process, particularly the IsHoliday column that appeared in both features and sales datasets.

Through systematic visualization including histograms, count plots, time series analysis, and scatter matrix plots, several important patterns emerged. The temperature data showed clear seasonal variations when plotted over time, while store types demonstrated relatively balanced distribution across the dataset. The scatter matrix revealed relationships between economic indicators, with CPI and unemployment showing some correlation as expected in economic data. Holiday distribution analysis showed the dataset contains both holiday and non-holiday periods, providing important context for sales forecasting.

The data cleaning process focused primarily on handling missing values in the markdown columns, where missing values were appropriately filled with zeros since absence of promotional markdowns indicates no promotional activity. Missing value visualization through heatmaps confirmed that missingness was concentrated in promotional features rather than core operational data. Outlier detection through box plots identified some extreme values, leading to standardization of numeric features to ensure consistent scaling across all variables.

Correlation analysis revealed moderate relationships between economic indicators and weaker correlations among other features, suggesting no severe multicollinearity issues that would complicate modeling. The creation of a new MarkDownTotal feature by summing all individual markdown columns provides a comprehensive measure of promotional intensity, potentially offering better predictive power than individual markdown features. This engineered feature showed interesting distribution patterns and maintained reasonable correlation with other features.

The preprocessing pipeline successfully converted all categorical variables to numeric format, with store types mapped to numeric codes and boolean holiday indicators converted to binary values. Data standardization ensures that features with different scales can be effectively used in machine learning models. The final dataset maintains temporal structure necessary for sales forecasting while providing comprehensive store-level and time-varying features that should enable robust predictive modeling.

In all, this analysis establishes a solid foundation for the subsequent modeling phase, with clean, well-understood data ready for sales forecasting applications.