分析結果

ざっとデータ確認

- 各csv毎に統計量/欠損値確認
  - train.csv
    - 統計量

In [2]:
```
train_data = pd.read_csv("data/train.csv")
train_data.describe(include='all')
```

Out[2]:

	Store	Dept	Date	Weekly_Sales	IsHoliday
count	421570.000000	421570.000000	421570	421570.000000	421570
unique	NaN	NaN	143	NaN	2
top	NaN	NaN	2011-12-23	NaN	False
freq	NaN	NaN	3027	NaN	391909
mean	22.200546	44.260317	NaN	15981.258123	NaN
std	12.785297	30.492054	NaN	22711.183519	NaN
min	1.000000	1.000000	NaN	-4988.940000	NaN
25%	11.000000	18.000000	NaN	2079.650000	NaN
50%	22.000000	37.000000	NaN	7612.030000	NaN
75%	33.000000	74.000000	NaN	20205.852500	NaN
max	45.000000	99.000000	NaN	693099.360000	NaN

    - 欠損値確認

In [3]:
```
train_data.isnull().sum()
```

Out[3]:
```
Store         0
Dept          0
Date          0
Weekly_Sales  0
IsHoliday     0
dtype: int64
```

  - stores.csv
    - 統計量

In [4]:
```
store_data = pd.read_csv('data/stores.csv')
store_data.describe(include='all')
```

Out[4]:

	Store	Type	Size
count	45.000000	45	45.000000
unique	NaN	3	NaN
top	NaN	A	NaN
freq	NaN	22	NaN
mean	23.000000	NaN	130287.600000
std	13.133926	NaN	63825.271991
min	1.000000	NaN	34875.000000
25%	12.000000	NaN	70713.000000
50%	23.000000	NaN	126512.000000
75%	34.000000	NaN	202307.000000
max	45.000000	NaN	219622.000000

    - 欠損値確認

In [5]:
```
store_data.isnull().sum()
```

Out[5]:
```
Store    0
Type     0
Size     0
dtype: int64
```

  - features.csv

- 統計量

```
In [6]: features_data = pd.read_csv("data/features.csv")
features_data.describe(include='all')
```

Out[6]:

	Store	Date	Temperature	Fuel_Price	MarkDown1	MarkDown2	MarkDown3	MarkDown4	MarkDown5	CPI	Unemployment
count	8190.000000	8190	8190.000000	8190.000000	4032.000000	2921.000000	3613.000000	3464.000000	4050.000000	7605.000000	7605.000000
unique	NaN	182	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
top	NaN	2010-02-05	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
freq	NaN	45	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
mean	23.000000	NaN	59.356198	3.405992	7032.371786	3384.176594	1760.100180	3292.935886	4132.216422	172.460809	7.826821
std	12.987966	NaN	18.678607	0.431337	9262.747448	8793.583016	11276.462208	6792.329861	13086.690278	39.738346	1.877259
min	1.000000	NaN	-7.290000	2.472000	-2781.450000	-265.760000	-179.260000	0.220000	-185.170000	126.064000	3.684000
25%	12.000000	NaN	45.902500	3.041000	1577.532500	68.880000	6.600000	304.687500	1440.827500	132.364839	6.634000
50%	23.000000	NaN	60.710000	3.513000	4743.580000	364.570000	36.260000	1176.425000	2727.135000	182.764003	7.806000
75%	34.000000	NaN	73.880000	3.743000	8923.310000	2153.350000	163.150000	3310.007500	4832.555000	213.932412	8.567000
max	45.000000	NaN	101.950000	4.468000	103184.980000	104519.540000	149483.310000	67474.850000	771448.100000	228.976456	14.313000

- 欠損値確認

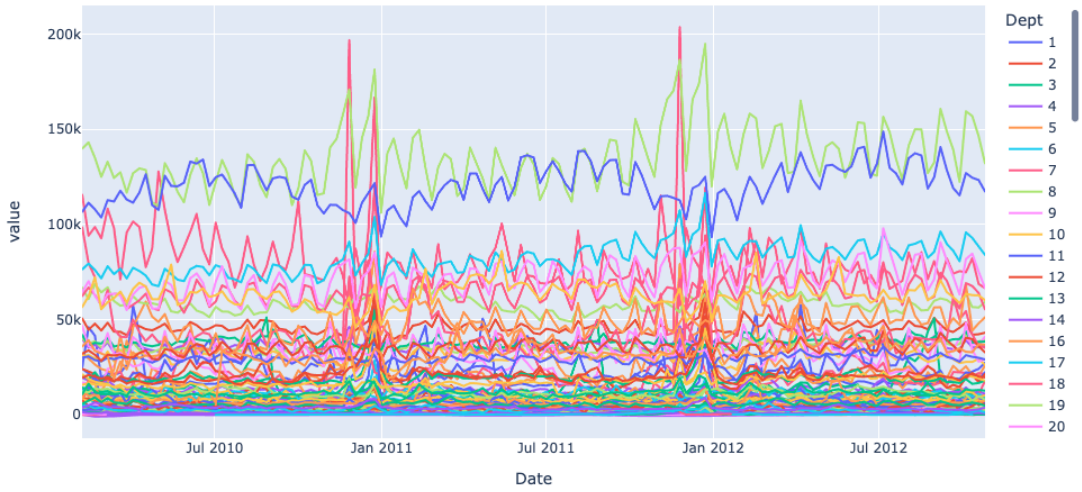
```
In [7]: features_data.isnull().sum()
```

```
Out[7]: Store          0
Date           0
Temperature    0
Fuel_Price     0
MarkDown1      4158
MarkDown2      5269
MarkDown3      4577
MarkDown4      4726
MarkDown5      4140
CPI            585
Unemployment   585
IsHoliday      0
dtype: int64
```

stores.csvとfeatures.csvをtrain.csvにjoinして部門毎売上のトレンド確認

- Store毎に確認すると
  - 4つの休日の中で、ほとんどの店舗で感謝財の週が一番売り上げの変動が見て取れる
    - IsHolidayとは別にIsThanksgivingのフラグを特徴量として追加したほうが良さそう
  - しかし、部門毎でみると偏りは感じる

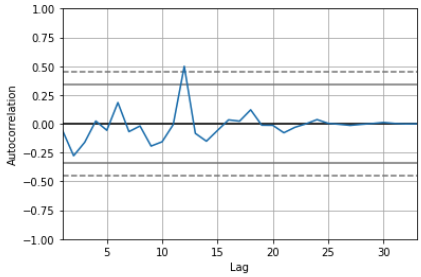
```
In [10]: store_one = joined_data.query('Store == 1')
fig = px.line(store_one, x='Date', y=['Weekly_Sales'], color='Dept')
fig.show()
```



- 周期性がありそうか確認する
  - 月単位で売上をまとめて(平均とる)周期性を確認した
    - ラグ12地点の相関係数の値は他の地点と比較して大きいのでDeteはMonthとして特徴量に加える

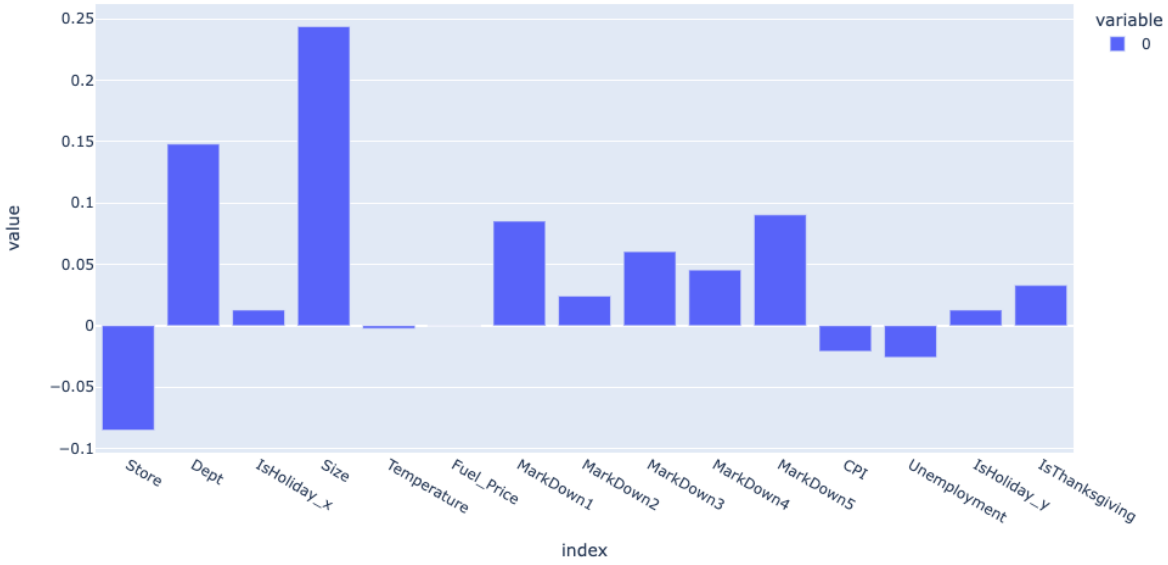
```
In [14]: from pandas.plotting import autocorrelation_plot

         joined_data = joined_data.assign(yearMonth = joined_data['Date'].str[0:7])
         autocorrelation_plot(joined_data.groupby('yearMonth').mean()['Weekly_Sales'])
Out[14]: <AxesSubplot:xlabel='Lag', ylabel='Autocorrelation'>
```



IsThanksgivingを含む情報と部門毎売上の相関を確認した

- MarkDown1 ~ 5とCPI、Unemploymentは相関が強そうなので細かく見てみる



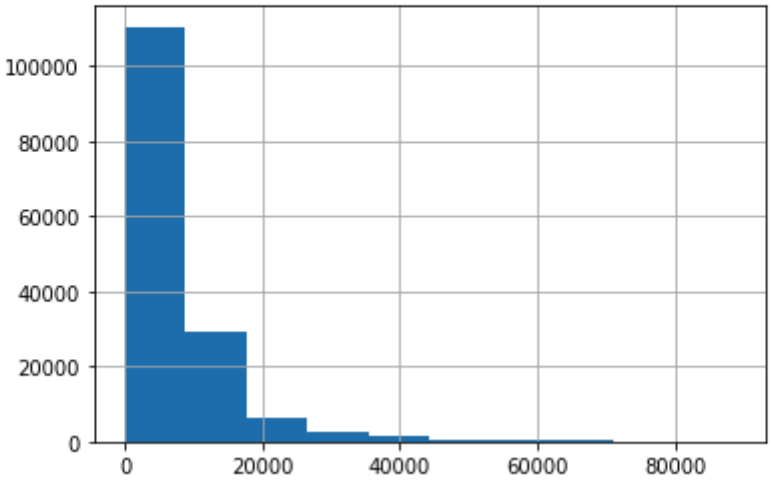
MarkDown1 ~ 5について

- 分布を確認
  - 分布がだいぶ偏っている
  - 2011年11月以降にしか使えない

- そのため、今回は特徴量として扱うのは難しそう

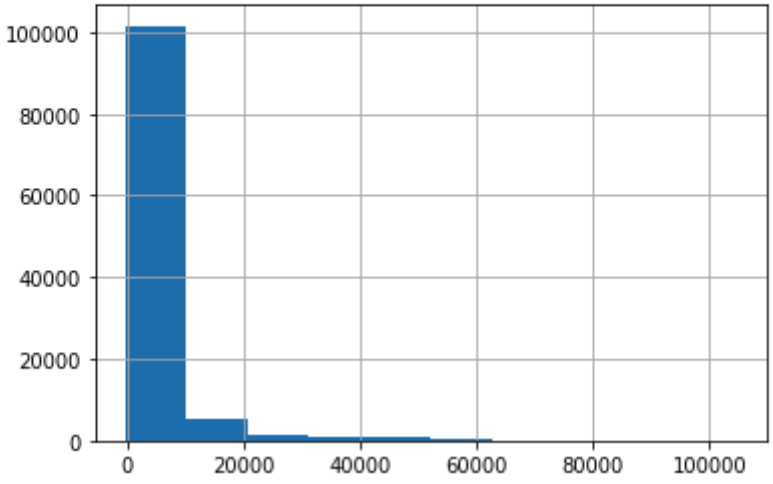
In [16]:
```
joined_data['MarkDown1'].hist()
```

Out[16]: <AxesSubplot:>



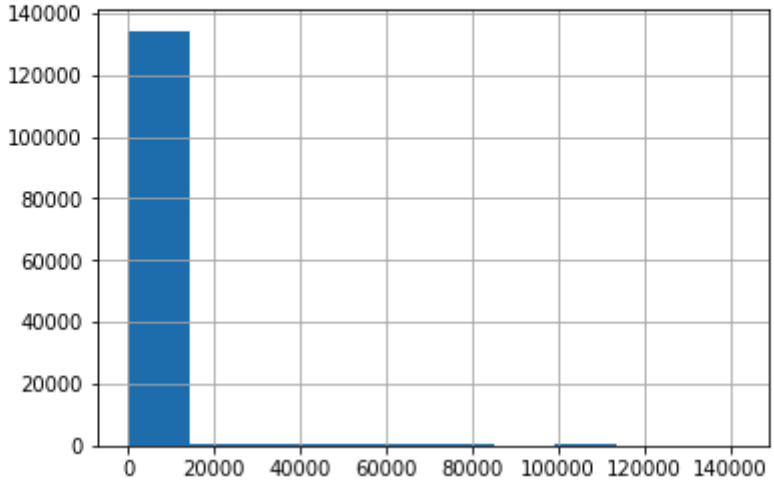
In [17]:
```
joined_data['MarkDown2'].hist()
```

Out[17]: <AxesSubplot:>



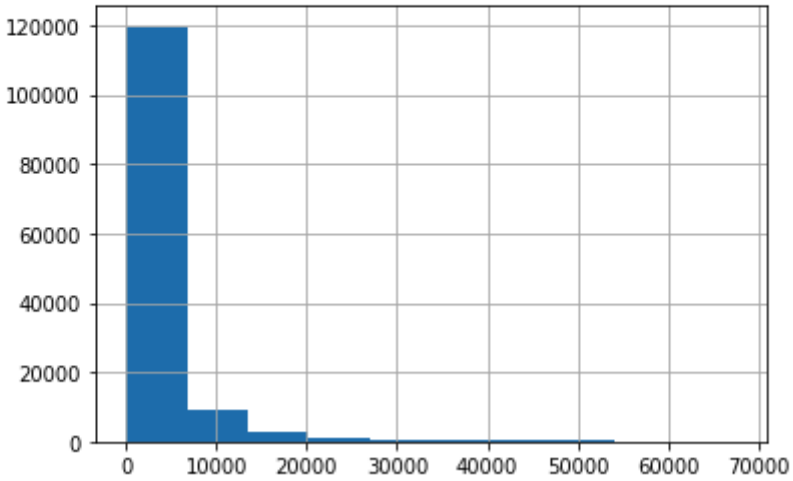
In [18]:
```
joined_data['MarkDown3'].hist()
```

Out[18]: <AxesSubplot:>



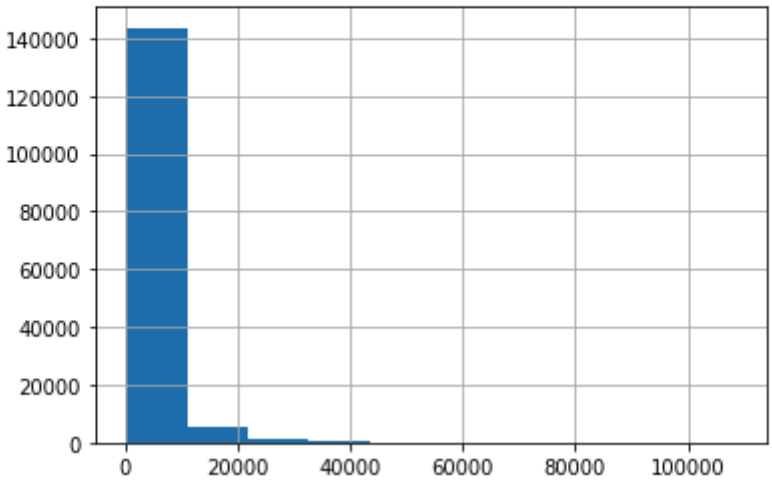
In [19]:
```
joined_data['MarkDown4'].hist()
```

Out[19]: <AxesSubplot:>



In [20]:
```
joined_data['MarkDown5'].hist()
```

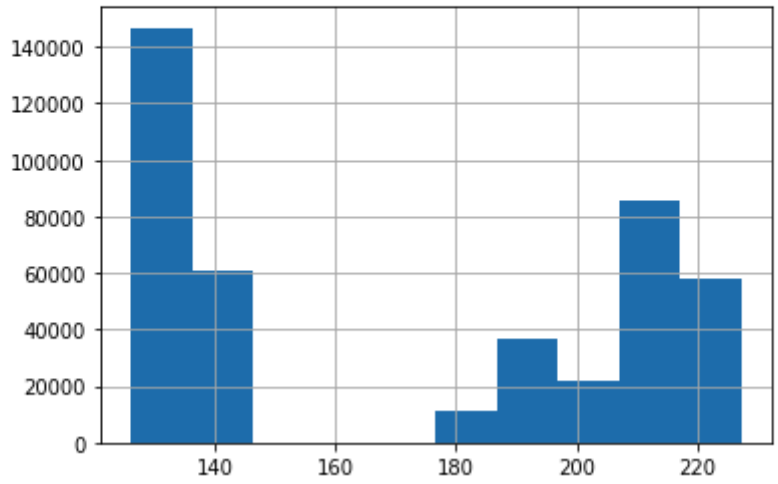
Out[20]: <AxesSubplot:>



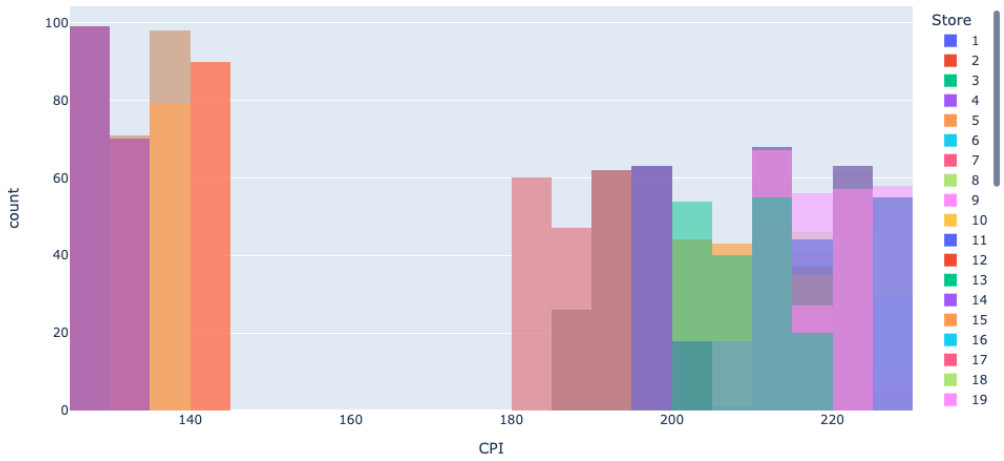
CPIについて

- 分布を確認
    - 分布が2峰型のためグルーピングが異なっている可能性がある
        - そのため、Store毎に分布を確認すると2のグループに分けられる
    - 欠損値について
        - 今回は、Store毎に平均を使用することで補完する

```
In [21]: joined_data['CPI'].hist()
Out[21]: <AxesSubplot:>
```




```
In [22]: px.histogram(features_data, x='CPI', color='Store', barmode='overlay')
```

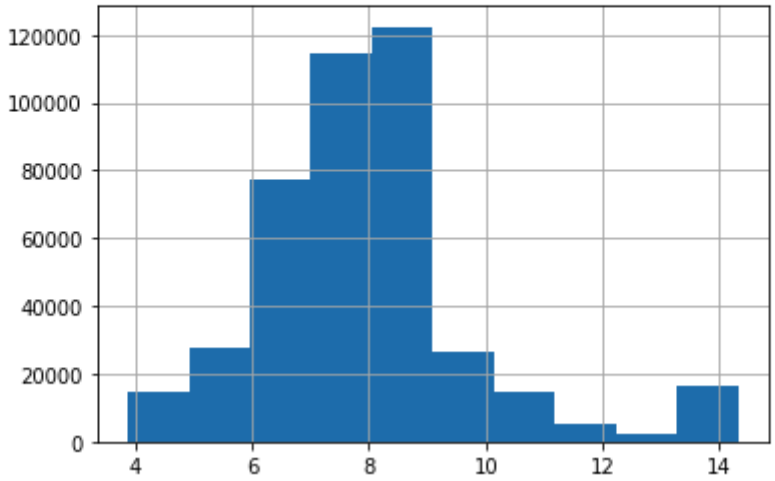


Unemploymentについて

- 分布確認
    - 少し偏りはあるが、大体正規分布っぽい形になっている
- 欠損値について

- 今回は、全体の平均値を使用することで補完する

```
In [23]: joined_data['Unemployment'].hist()
Out[23]: <AxesSubplot:>
```



使用する特徴量について

- 今回は以情報を特徴量として扱う
  - Store
  - Dept
  - Month
  - IsHoliday
  - Type
  - Size
  - CPI
  - Unemployment
  - IsThanksgiving (感謝祭の週かどうか)

考察

- 売上に対する予測なので、地域差や時期、イベント等が起因することが予測される
  - 実際、IsHokidayを確認するとIsHokiday=Trueの際に売上が変動している店舗/部門が確認された
  - また店舗毎で部門売上のトレンドが異なっておりそれぞれの特色があるように見える
  - 与えられた情報に関しても、店舗(ここは地域差なのか)毎で異なる傾向を確認できた
- そのため、使用する特徴量であげた情報を使用することで売上予測が特徴を捉えた予測ができそうと考える