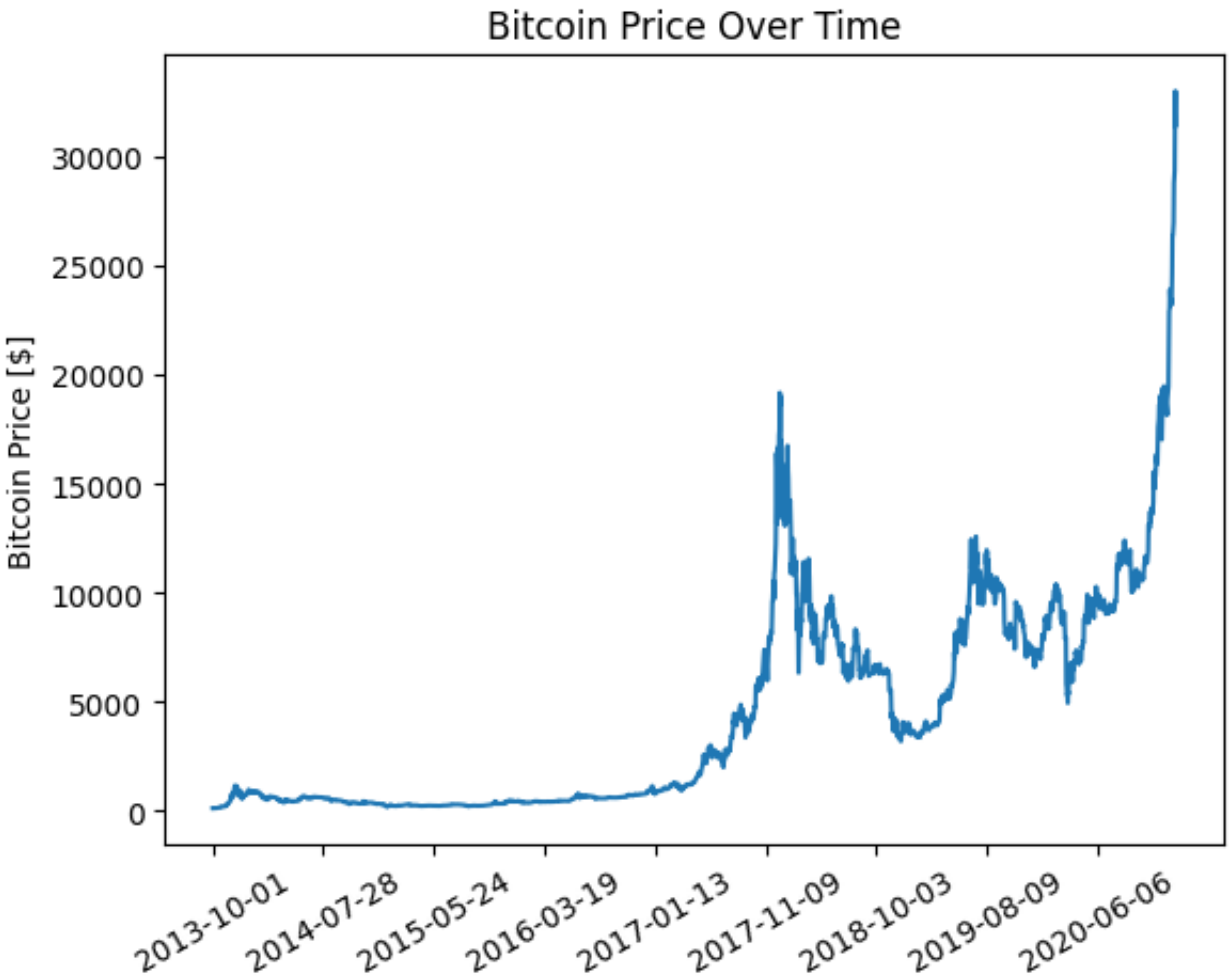
In [1]:
```
import matplotlib.pyplot as plt
from torch import nn
import numpy as np
```

In [17]:
```
import urllib.request, csv, io
from collections import defaultdict
import matplotlib.pyplot as plt

def load_csv_from_url(url):
    with urllib.request.urlopen(url) as response:
        csv_data = response.read().decode('utf-8')
        csv_file = io.StringIO(csv_data)
        csv_reader = csv.DictReader(csv_file)

        # Store all data into a dictionary
        data_dict = defaultdict(list)
        for row in csv_reader:
            for key, value in row.items():
                data_dict[key].append(value)

        return data_dict

# Load data from URL
bitcoin_url = 'https://raw.githubusercontent.com/RDeconomist/observatory/mai
bitcoin_data = load_csv_from_url(bitcoin_url)

# Extract date and price data
date = bitcoin_data['Date']
price = [float(v) for v in bitcoin_data['Closing Price (USD)']]

## Part (a): Find min and max prices and corresponding dates
min_price = min(price)
max_price = max(price)

min_index = price.index(min_price)
max_index = price.index(max_price)

print(f'Date range: {date[0]} to {date[-1]}')
print(f'Min 1BTC = ${min_price:.2f}')
print(f'Min price occurred on {date[min_index]}')
print(f'Max 1BTC = ${max_price:.2f}')
print(f'Max price occurred on {date[max_index]}')

# Plot price data
plt.plot(price)
plt.ylabel('Bitcoin Price [$]')
plt.gca().set_xticks(range(0, len(date), 300))
plt.gca().set_xticklabels(date[::300], rotation=30)
plt.title('Bitcoin Price Over Time')
plt.show()
```

```
Date range: 2013-10-01 to 2021-01-04
Min 1BTC = $108.58
Min price occurred on 2013-10-03
Max 1BTC = $33002.54
Max price occurred on 2021-01-03
```



In [18]:
```
len(price)
```

Out[18]: 2613

In [19]:
```
import torch
from torch.utils.data import Dataset, DataLoader

class TimeSeriesDataset(Dataset):
    def __init__(self, time_series, sequence_length):
        self.time_series = time_series
        self.sequence_length = sequence_length

    def __len__(self):
        # The total number of sequences in the time_series
        return len(self.time_series) - self.sequence_length

    def __getitem__(self, idx):
        # Extract the time sequence
```

```
        sequence = self.time_series[idx : (idx + self.sequence_length)]
        # Convert the sequence to a PyTorch tensor
        sequence = torch.tensor(sequence)
        return sequence

# Part (b)
print('=== Dataset ===')
dataset = TimeSeriesDataset(price, 30)
target_index = date.index('2017-12-01')
p = dataset[target_index]
print(f"Prices starting on 2017-12-01:\n ", p)
```

```
=== Dataset ===
Prices starting on 2017-12-01:
  tensor([ 9706.1035, 10923.2012, 10973.5439, 11382.2090, 11597.2314, 12230.
3652,
        13734.5195, 16403.4219, 15732.0996, 13152.5283, 16299.2979, 16374.89
75,
        16678.2871, 16246.1387, 17221.6504, 17436.5977, 19166.9785, 18640.26
17,
        18984.7676, 16862.5703, 16925.0176, 14182.4648, 14694.5820, 14103.21
88,
        13387.3486, 14652.3672, 15846.0742, 14362.4033, 14867.5723, 13643.69
53])
```

Linear Regression

Let's fit a linear regression model to predict the bitcoin price on the 10th day from the previous 9 days.

In [50]:
```
def loss(pred, y):
    """Mean Absolute Error loss"""
    return (pred - y).abs().mean()

# Part (d)
def train(model, loss, dataloader, optimizer):
    """Helper function to train our model."""
    total_error = 0.
    for it, sequences in enumerate(dataloader):
        # Prepare model inputs and targets
        price_history = sequences[:, :9]
        target = sequences[:, -1]
        # Compute model predictions
        pred = model(price_history)
        # Compute the loss
        l = loss(pred, target)
        total_error += l.item()
        # Update the weights
        optimizer.zero_grad()  # Clear the gradients
        l.backward()  # Backpropagation
        optimizer.step()  # Update the model weights
    return total_error / len(dataloader)
```

```python
def fit(model, loss, dataloader, epochs=30):
    optimizer = torch.optim.Adam(model.parameters(), lr=lr)
    for ep in range(epochs):
        err = train(model, loss, dataloader, optimizer)
        print(f"[Ep{ep}] Error {err:.3f}")

price_history_len = 10
lr = 0.0005
batch_size = 32
epochs = 300

# Part (c)
model = nn.Linear(in_features=9, out_features=1)
dataset = TimeSeriesDataset(price, price_history_len)
dataloader = DataLoader(dataset, batch_size=batch_size, shuffle=True, drop_l
fit(model, loss, dataloader, epochs=epochs)
```

```
[Ep0] Error 5672.930
[Ep1] Error 4326.833
[Ep2] Error 4026.067
[Ep3] Error 4023.173
[Ep4] Error 4012.734
[Ep5] Error 4025.728
[Ep6] Error 4017.178
[Ep7] Error 4017.781
[Ep8] Error 4023.927
[Ep9] Error 4011.991
[Ep10] Error 4007.750
[Ep11] Error 4009.470
[Ep12] Error 4015.672
[Ep13] Error 4007.969
[Ep14] Error 4006.800
[Ep15] Error 4001.535
[Ep16] Error 4012.020
[Ep17] Error 4015.863
[Ep18] Error 4009.041
[Ep19] Error 4006.799
[Ep20] Error 4016.354
[Ep21] Error 4004.125
[Ep22] Error 4012.177
[Ep23] Error 4011.736
[Ep24] Error 4007.859
[Ep25] Error 4014.734
[Ep26] Error 4011.939
[Ep27] Error 4009.344
[Ep28] Error 4018.528
[Ep29] Error 4012.901
[Ep30] Error 4022.619
[Ep31] Error 4014.874
[Ep32] Error 4003.760
[Ep33] Error 4014.397
[Ep34] Error 4009.345
[Ep35] Error 4008.765
```

[Ep36] Error 4004.516
[Ep37] Error 4018.724
[Ep38] Error 4000.186
[Ep39] Error 4009.327
[Ep40] Error 4025.034
[Ep41] Error 4019.089
[Ep42] Error 4025.503
[Ep43] Error 4013.457
[Ep44] Error 4004.051
[Ep45] Error 4023.152
[Ep46] Error 4001.947
[Ep47] Error 4003.005
[Ep48] Error 4012.513
[Ep49] Error 4014.222
[Ep50] Error 4005.332
[Ep51] Error 4009.719
[Ep52] Error 4013.887
[Ep53] Error 4005.753
[Ep54] Error 4007.430
[Ep55] Error 4017.226
[Ep56] Error 4017.778
[Ep57] Error 4005.493
[Ep58] Error 4018.490
[Ep59] Error 4008.066
[Ep60] Error 4020.161
[Ep61] Error 4013.537
[Ep62] Error 4011.636
[Ep63] Error 4009.193
[Ep64] Error 4003.016
[Ep65] Error 4011.214
[Ep66] Error 4012.506
[Ep67] Error 4012.420
[Ep68] Error 3997.517
[Ep69] Error 4015.289
[Ep70] Error 4007.477
[Ep71] Error 4009.981
[Ep72] Error 4017.471
[Ep73] Error 4007.192
[Ep74] Error 3998.331
[Ep75] Error 4000.123
[Ep76] Error 4006.057
[Ep77] Error 4017.277
[Ep78] Error 3990.929
[Ep79] Error 4015.916
[Ep80] Error 4005.934
[Ep81] Error 4002.386
[Ep82] Error 4010.037
[Ep83] Error 4000.030
[Ep84] Error 4016.793
[Ep85] Error 4000.927
[Ep86] Error 3993.003
[Ep87] Error 4003.857
[Ep88] Error 4001.864

[Ep89] Error 4016.669
[Ep90] Error 4001.480
[Ep91] Error 3990.692
[Ep92] Error 4006.216
[Ep93] Error 4003.682
[Ep94] Error 4008.447
[Ep95] Error 4016.682
[Ep96] Error 4010.924
[Ep97] Error 4013.372
[Ep98] Error 4008.933
[Ep99] Error 4015.248
[Ep100] Error 4014.768
[Ep101] Error 4002.968
[Ep102] Error 4007.541
[Ep103] Error 4004.826
[Ep104] Error 3996.546
[Ep105] Error 3995.413
[Ep106] Error 4005.060
[Ep107] Error 4011.560
[Ep108] Error 4009.396
[Ep109] Error 4014.831
[Ep110] Error 4004.589
[Ep111] Error 4004.678
[Ep112] Error 4003.331
[Ep113] Error 4015.426
[Ep114] Error 4008.904
[Ep115] Error 4011.147
[Ep116] Error 4012.111
[Ep117] Error 4013.792
[Ep118] Error 4008.497
[Ep119] Error 4009.190
[Ep120] Error 4008.168
[Ep121] Error 3987.357
[Ep122] Error 4005.607
[Ep123] Error 4003.838
[Ep124] Error 3999.006
[Ep125] Error 4010.674
[Ep126] Error 4009.687
[Ep127] Error 4014.294
[Ep128] Error 4008.170
[Ep129] Error 4008.394
[Ep130] Error 4011.883
[Ep131] Error 4009.037
[Ep132] Error 4019.422
[Ep133] Error 4009.023
[Ep134] Error 4009.512
[Ep135] Error 3999.428
[Ep136] Error 4000.038
[Ep137] Error 4001.156
[Ep138] Error 3999.265
[Ep139] Error 4001.729
[Ep140] Error 4006.884
[Ep141] Error 3997.098

[Ep142] Error 3992.594
[Ep143] Error 4014.823
[Ep144] Error 4005.738
[Ep145] Error 4008.745
[Ep146] Error 4006.663
[Ep147] Error 4006.801
[Ep148] Error 3989.879
[Ep149] Error 4003.389
[Ep150] Error 4016.525
[Ep151] Error 3996.356
[Ep152] Error 4006.745
[Ep153] Error 4000.695
[Ep154] Error 4005.802
[Ep155] Error 4004.958
[Ep156] Error 4004.121
[Ep157] Error 4000.693
[Ep158] Error 4004.791
[Ep159] Error 4020.510
[Ep160] Error 4003.457
[Ep161] Error 4010.725
[Ep162] Error 3999.930
[Ep163] Error 4010.641
[Ep164] Error 4008.764
[Ep165] Error 4010.384
[Ep166] Error 3995.451
[Ep167] Error 4011.055
[Ep168] Error 4013.479
[Ep169] Error 4012.821
[Ep170] Error 4009.077
[Ep171] Error 4000.572
[Ep172] Error 4000.016
[Ep173] Error 4006.055
[Ep174] Error 3994.960
[Ep175] Error 4015.124
[Ep176] Error 3995.031
[Ep177] Error 3997.256
[Ep178] Error 4007.994
[Ep179] Error 4011.761
[Ep180] Error 4001.469
[Ep181] Error 4019.063
[Ep182] Error 4011.504
[Ep183] Error 4000.795
[Ep184] Error 4005.889
[Ep185] Error 3984.675
[Ep186] Error 4000.809
[Ep187] Error 4013.685
[Ep188] Error 4011.894
[Ep189] Error 4009.947
[Ep190] Error 4008.137
[Ep191] Error 4001.491
[Ep192] Error 3992.311
[Ep193] Error 4015.977
[Ep194] Error 4008.533

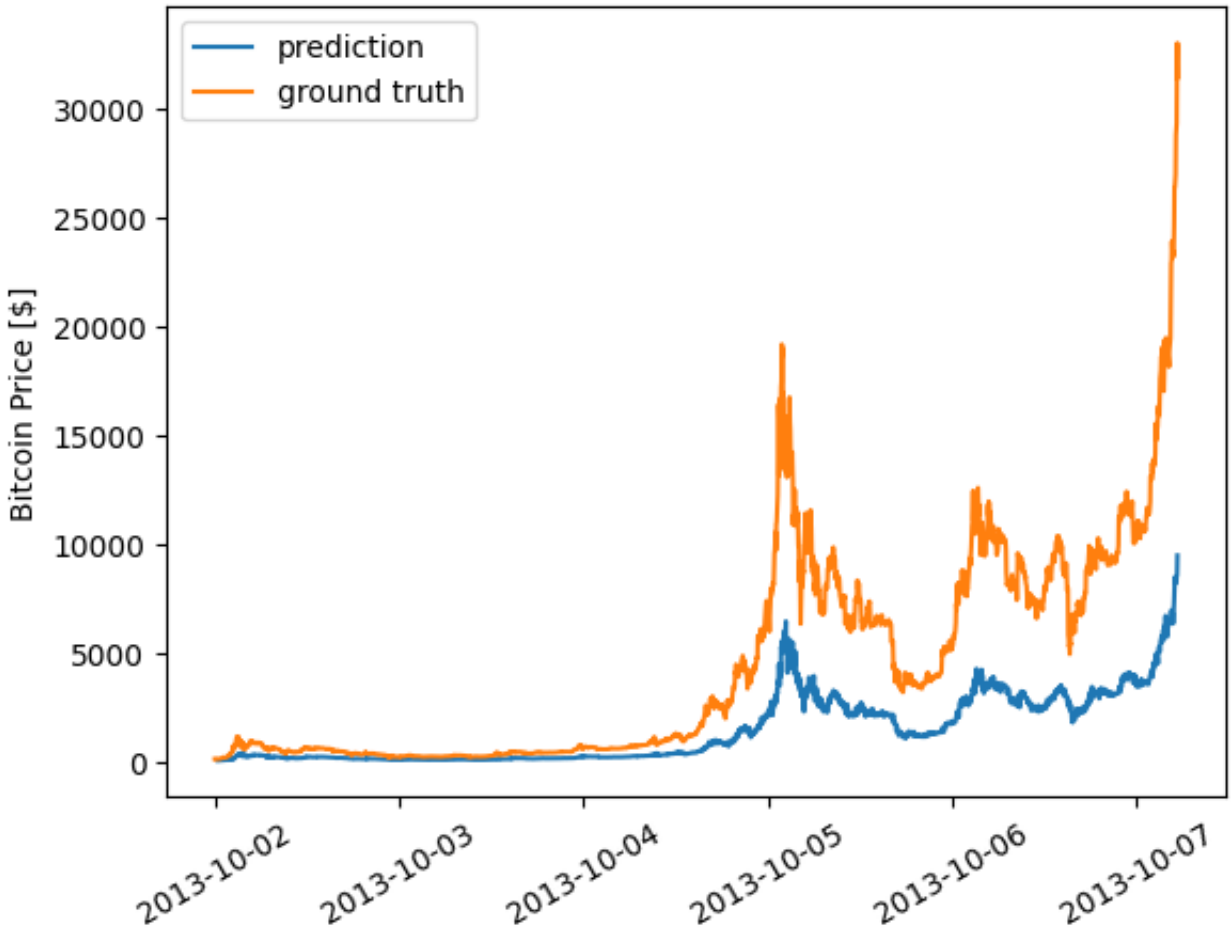
[Ep195] Error 4000.932
[Ep196] Error 4011.143
[Ep197] Error 4001.531
[Ep198] Error 3991.406
[Ep199] Error 4012.446
[Ep200] Error 4013.678
[Ep201] Error 4004.006
[Ep202] Error 3993.165
[Ep203] Error 3991.844
[Ep204] Error 4009.083
[Ep205] Error 4006.484
[Ep206] Error 4001.786
[Ep207] Error 4010.974
[Ep208] Error 4002.607
[Ep209] Error 4004.366
[Ep210] Error 3999.993
[Ep211] Error 4004.660
[Ep212] Error 3993.592
[Ep213] Error 4001.481
[Ep214] Error 3993.297
[Ep215] Error 4011.499
[Ep216] Error 4006.869
[Ep217] Error 4003.046
[Ep218] Error 3995.307
[Ep219] Error 4013.758
[Ep220] Error 4016.122
[Ep221] Error 4009.977
[Ep222] Error 4002.832
[Ep223] Error 3995.972
[Ep224] Error 3999.145
[Ep225] Error 4012.381
[Ep226] Error 4003.680
[Ep227] Error 4013.086
[Ep228] Error 3999.783
[Ep229] Error 3988.192
[Ep230] Error 4002.762
[Ep231] Error 3993.980
[Ep232] Error 4006.652
[Ep233] Error 4001.170
[Ep234] Error 3999.893
[Ep235] Error 4001.840
[Ep236] Error 3993.693
[Ep237] Error 3994.450
[Ep238] Error 3999.538
[Ep239] Error 4002.430
[Ep240] Error 4004.724
[Ep241] Error 4000.314
[Ep242] Error 3997.678
[Ep243] Error 4003.378
[Ep244] Error 3999.946
[Ep245] Error 3989.475
[Ep246] Error 4000.301
[Ep247] Error 4004.872

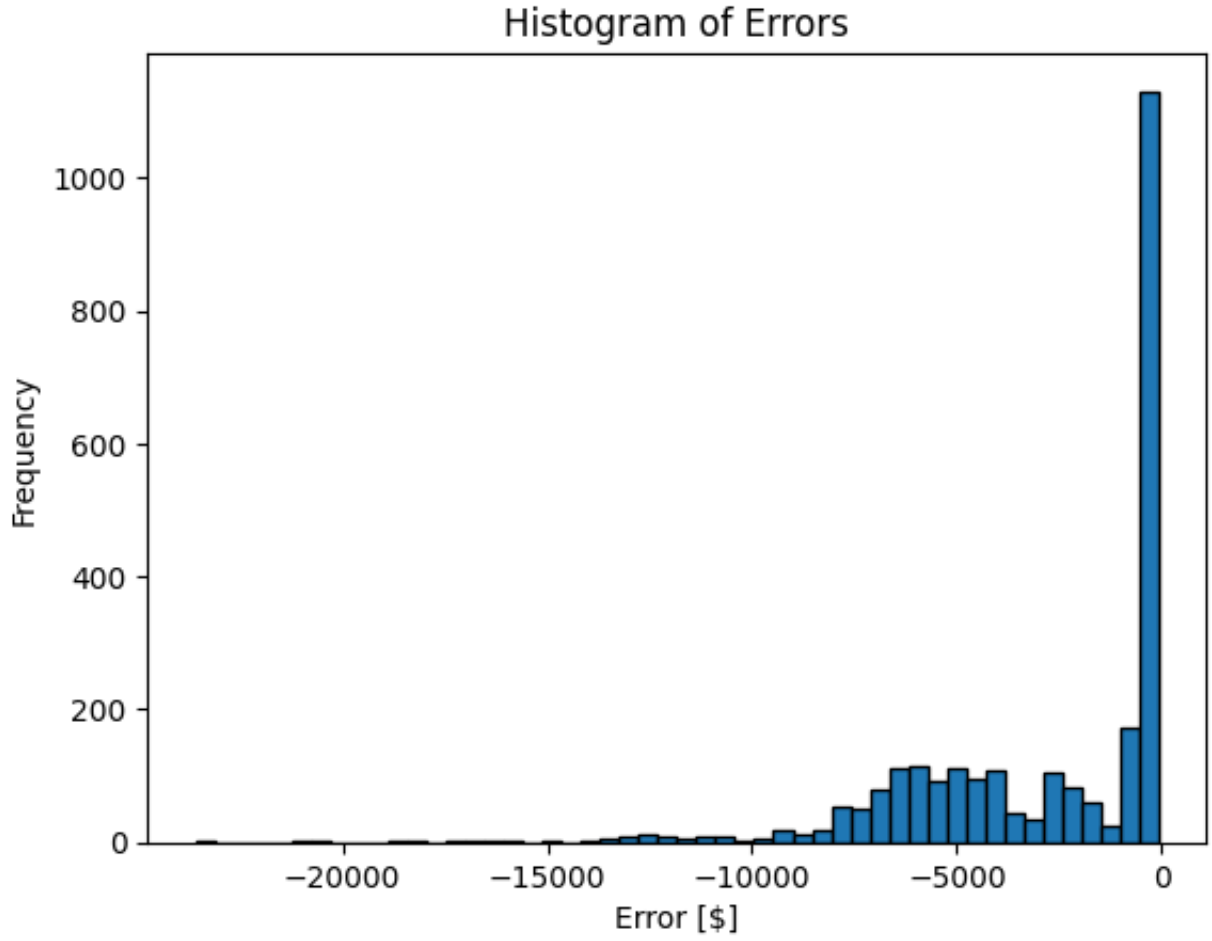
```
[Ep248] Error 4004.137
[Ep249] Error 4010.934
[Ep250] Error 3987.594
[Ep251] Error 3993.017
[Ep252] Error 4012.330
[Ep253] Error 4006.313
[Ep254] Error 4001.463
[Ep255] Error 3983.080
[Ep256] Error 4004.647
[Ep257] Error 4002.740
[Ep258] Error 4002.848
[Ep259] Error 4008.582
[Ep260] Error 4001.715
[Ep261] Error 4002.276
[Ep262] Error 4011.944
[Ep263] Error 3997.368
[Ep264] Error 3998.225
[Ep265] Error 4005.022
[Ep266] Error 3985.296
[Ep267] Error 4009.389
[Ep268] Error 3997.747
[Ep269] Error 3996.767
[Ep270] Error 4002.229
[Ep271] Error 4009.435
[Ep272] Error 4014.010
[Ep273] Error 4009.173
[Ep274] Error 3998.882
[Ep275] Error 3994.030
[Ep276] Error 4005.278
[Ep277] Error 3984.612
[Ep278] Error 4001.552
[Ep279] Error 4002.377
[Ep280] Error 4001.069
[Ep281] Error 3989.337
[Ep282] Error 3999.715
[Ep283] Error 3991.132
[Ep284] Error 4013.962
[Ep285] Error 4016.554
[Ep286] Error 4000.236
[Ep287] Error 3997.511
[Ep288] Error 3997.545
[Ep289] Error 4007.959
[Ep290] Error 4012.327
[Ep291] Error 4003.569
[Ep292] Error 4006.893
[Ep293] Error 4011.753
[Ep294] Error 3985.736
[Ep295] Error 4007.718
[Ep296] Error 4006.861
[Ep297] Error 3996.841
[Ep298] Error 3997.768
[Ep299] Error 3999.303
```

In [55]:
```
with torch.no_grad():
```

```
    predictions, errors = [], []
    for i in range(len(dataset)):
        sequence = dataset[i]
        past, price_gt = sequence[0:-1], sequence[-1]
        price_pred = model(past)

        err = price_pred - price_gt

        errors.append(err.item())
        predictions.append(price_pred.item())

plt.plot([None]*9+predictions, label='prediction')
plt.plot(price, label='ground truth')
plt.ylabel('Bitcoin Price [$]')
plt.gca().set_xticklabels(date, rotation=30)
plt.legend()
plt.show()

plt.hist(errors, bins=50, edgecolor='black')
plt.xlabel('Error [$]')
plt.ylabel('Frequency')
plt.title('Histogram of Errors')
plt.show()
```

```
/var/folders/tn/v9tpvrrs4qgdbw0xd1q0l8qh0000gn/T/ipykernel_30220/1442675984.
py:16: UserWarning: set_ticklabels() should only be used with a fixed number
of ticks, i.e. after set_ticks() or using a FixedLocator.
  plt.gca().set_xticklabels(date, rotation=30)
```


prediction
ground truth
Bitcoin Price [$]
30000
25000
20000
15000
10000
5000
0
2013-10-02
2013-10-03
2013-10-04
2013-10-05
2013-10-06
2013-10-07


Histogram of Errors
Frequency
1000
800
600
400
200
0
−20000
−15000
−10000
−5000
0
Error [$]

In [6]: