ANALYSIS OF DATA FROM BLUEPRINT FOREX APP

By Wofai Alfred Eyong

Data Analyst; Blueprint Fintech Solutions ltd.

Introduction

Blueprint Fintech Solutions ltd is a Lagos, Nigeria-based company in the foreign exchange market; a global marketplace where banks, institutions and investors, place trades and calls on different currency pairs. This company created its first software application: THE BLUEPRINT FOREX APP, which is a place where subscribers can receive trading calls, signals and information to aid trading the financial market. With over 300 subscribers after the app launch, trade calls were given over an 80 day period. All generated information from the company spanning across several variables was collated to curate the 2 datasets which were merged and analysed. The dataset contains information on about 15 variables, all of which are focused on the trade calls given from the app and their outcomes. The variables analysed include:

Date: Date of record creation in table 1. It corresponds to the open date.

Pairs: Currency pairs traded.

Trade: Trade calls (Buy or Sell).

Open: Date trade was opened.

Close: Date trade was closed.

result: Trade outcome (Successful or Unsuccessful).

lot_size: Lot size used for trade.

no_of_pips_won: Number of pips won after trade was closed.

no_of_pips_lost: Number of pips lost after trade was closed.

amt_won: Amount in dollars won after trade was closed.

amt_lost: Amount in dollars lost after trade was closed.

Index_day: Trade day count (From monday to friday only)

Date: Date of record creation in table 2.

no_of_trades_entered: Number of trades entered on a particular day.

no_of_trades_closed: Number of trades closed on a particular day.

In [3]:
```
#importing modules
import pandas as pd
import numpy as np
```

```
from matplotlib import pyplot as plt
import seaborn as sns

%matplotlib inline
```

Data Assessment

Step 1: Load the dataset

In this step, I loaded the dataset using the pandas read_csv function. After loading the dataset, I confirm loading status by running the pandas head function

In [4]:
```
#loading the first dataset
df1 = pd.read_csv('Blueprint data.csv')
df1.head()
```

Out[4]:

	Date	Pairs	Trade	open	close	result	lot_size	no_of_pips_won	no_of_pips_lost	ar
0	22/08/2022	GBPJPY	SELL	22/08/2022	23/08/2022	successful	0.01	34.2	0.0	
1	22/08/2022	AUDUSD	SELL	22/08/2022	22/08/2022	successful	0.01	16.8	0.0	
2	23/08/2022	GBPCHF	BUY	23/08/2022	23/08/2022	successful	0.01	33.2	0.0	
3	24/08/2022	AUDUSD	SELL	24/08/2022	25/08/2022	unsuccessful	0.01	0.0	39.0	
4	25/08/2022	GBPJPY	SELL	25/08/2022	26/08/2022	unsuccessful	0.01	0.0	15.9	

In [5]:
```
#loading the second dataset
df2 = pd.read_csv('Blueprint_data_count.csv')
df2.head()
```

Out[5]:

	Index_day	Date	no_of_trades_entered	no_of_trades_closed
0	1	22/08/2022	2	1
1	2	23/08/2022	1	2
2	3	24/08/2022	1	0
3	4	25/08/2022	1	1
4	5	26/08/2022	3	4

In [6]:
```
df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 73 entries, 0 to 72
Data columns (total 11 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   Date             73 non-null     object 
 1   Pairs            73 non-null     object 
 2   Trade            73 non-null     object 
 3   open             73 non-null     object 
 4   close            73 non-null     object 
 5   result           73 non-null     object 
 6   lot_size         73 non-null     float64
 7   no_of_pips_won   73 non-null     float64
 8   no_of_pips_lost  73 non-null     float64
 9   amt_won          73 non-null     float64
 10  amt_lost         73 non-null     float64
```

```
dtypes: float64(5), object(6)
memory usage: 6.4+ KB
```

In [7]:
```
df2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 80 entries, 0 to 79
Data columns (total 4 columns):
 #   Column                 Non-Null Count  Dtype 
---  ------                 --------------  ----- 
 0   Index_day              80 non-null     int64 
 1   Date                   80 non-null     object
 2   no_of_trades_entered   80 non-null     int64 
 3   no_of_trades_closed    80 non-null     int64 
dtypes: int64(3), object(1)
memory usage: 2.6+ KB
```

In [8]:
```
#full outer join to merge both datasets to a master dataset
df = pd.merge(df1,df2,on='Date',how='outer')
```

In [9]:
```
#view new dataset
df.head()
```

Out[9]:

	Date	Pairs	Trade	open	close	result	lot_size	no_of_pips_won	no_of_pips_lost	ar
0	22/08/2022	GBPJPY	SELL	22/08/2022	23/08/2022	successful	0.01	34.2	0.0	
1	22/08/2022	AUDUSD	SELL	22/08/2022	22/08/2022	successful	0.01	16.8	0.0	
2	23/08/2022	GBPCHF	BUY	23/08/2022	23/08/2022	successful	0.01	33.2	0.0	
3	24/08/2022	AUDUSD	SELL	24/08/2022	25/08/2022	unsuccessful	0.01	0.0	39.0	
4	25/08/2022	GBPJPY	SELL	25/08/2022	26/08/2022	unsuccessful	0.01	0.0	15.9	

In [10]:
```
#view new dataset info
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 113 entries, 0 to 112
Data columns (total 14 columns):
 #   Column                 Non-Null Count  Dtype  
---  ------                 --------------  -----  
 0   Date                   113 non-null    object 
 1   Pairs                  73 non-null     object 
 2   Trade                  73 non-null     object 
 3   open                   73 non-null     object 
 4   close                  73 non-null     object 
 5   result                 73 non-null     object 
 6   lot_size               73 non-null     float64
 7   no_of_pips_won         73 non-null     float64
 8   no_of_pips_lost        73 non-null     float64
 9   amt_won                73 non-null     float64
 10  amt_lost               73 non-null     float64
 11  Index_day              113 non-null    int64  
 12  no_of_trades_entered   113 non-null    int64  
 13  no_of_trades_closed    113 non-null    int64  
dtypes: float64(5), int64(3), object(6)
memory usage: 13.2+ KB
```

In [11]:
```
# pd.reset_option('all') to display all rows
pd.set_option('display.max_rows', None)
df.head(113)
```

Out[11]:

	Date	Pairs	Trade	open	close	result	lot_size	no_of_pips_won	no_of_pips_lost

0	22/08/2022	GBPJPY	SELL	22/08/2022	23/08/2022	successful	0.01	34.2	0.0
1	22/08/2022	AUDUSD	SELL	22/08/2022	22/08/2022	successful	0.01	16.8	0.0
2	23/08/2022	GBPCHF	BUY	23/08/2022	23/08/2022	successful	0.01	33.2	0.0
3	24/08/2022	AUDUSD	SELL	24/08/2022	25/08/2022	unsuccessful	0.01	0.0	39.0
4	25/08/2022	GBPJPY	SELL	25/08/2022	26/08/2022	unsuccessful	0.01	0.0	15.9
5	26/08/2022	GBPCHF	SELL	26/08/2022	26/08/2022	successful	0.01	19.7	0.0
6	26/08/2022	AUDUSD	SELL	26/08/2022	26/08/2022	successful	0.01	16.4	0.0
7	26/08/2022	GBPJPY	SELL	26/08/2022	26/08/2022	successful	0.01	9.4	0.0
8	30/08/2022	GBPJPY	SELL	30/08/2022	30/08/2022	successful	0.01	32.5	0.0
9	31/08/2022	GBPJPY	SELL	31/08/2022	02/09/2022	unsuccessful	0.01	0.0	70.9
10	01/09/2022	EURJPY	SELL	01/09/2022	02/09/2022	unsuccessful	0.01	0.0	79.3
11	06/09/2022	AUDUSD	SELL	06/09/2022	06/09/2022	successful	0.01	13.7	0.0
12	06/09/2022	XAUUSD	SELL	06/09/2022	06/09/2022	successful	0.01	21.9	0.0
13	06/09/2022	GBPCAD	SELL	06/09/2022	07/09/2022	successful	0.01	18.7	0.0
14	07/09/2022	GBPJPY	SELL	07/09/2022	08/09/2022	successful	0.01	18.4	0.0
15	07/09/2022	USDCHF	SELL	07/09/2022	07/09/2022	successful	0.01	30.8	0.0
16	08/09/2022	USDCHF	SELL	08/09/2022	08/09/2022	successful	0.01	4.4	0.0
17	08/09/2022	AUDUSD	SELL	08/09/2022	09/09/2022	unsuccessful	0.01	0.0	114.9
18	08/09/2022	EURUSD	SELL	08/09/2022	09/09/2022	unsuccessful	0.01	0.0	113.0
19	09/09/2022	EURUSD	SELL	09/09/2022	12/09/2022	unsuccessful	0.01	0.0	71.8
20	09/09/2022	GBPJPY	SELL	09/09/2022	16/09/2022	successful	0.01	45.3	0.0
21	09/09/2022	AUDUSD	SELL	09/09/2022	13/09/2022	successful	0.01	25.9	0.0
22	15/09/2022	XAUUSD	SELL	15/09/2022	15/09/2022	successful	0.01	24.5	0.0
23	15/09/2022	EURUSD	SELL	15/09/2022	19/09/2022	unsuccessful	0.01	0.0	3.6
24	16/09/2022	XAUUSD	SELL	16/09/2022	16/09/2022	successful	0.01	59.9	0.0
25	16/09/2022	GBPJPY	SELL	16/09/2022	21/09/2022	successful	0.01	41.6	0.0
26	20/09/2022	XAUUSD	SELL	20/09/2022	20/09/2022	successful	0.01	64.4	0.0
27	20/09/2022	XAUUSD	SELL	20/09/2022	20/09/2022	unsuccessful	0.01	0.0	2.1
28	21/09/2022	AUDCAD	SELL	21/09/2022	22/09/2022	unsuccessful	0.01	0.0	21.4
29	21/09/2022	XAUUSD	SELL	21/09/2022	21/09/2022	successful	0.01	39.9	0.0
30	23/09/2022	EURJPY	BUY	23/09/2022	28/09/2022	successful	0.01	32.2	0.0
31	23/09/2022	GBPJPY	BUY	23/09/2022	30/09/2022	successful	0.01	56.6	0.0
32	26/09/2022	XAUUSD	SELL	26/09/2022	21/10/2022	unsuccessful	0.01	0.0	5.3
33	29/09/2022	AUDUSD	SELL	29/09/2022	30/09/2022	successful	0.01	63.5	0.0
34	04/10/2022	AUDJPY	SELL	04/10/2022	04/10/2022	successful	0.01	23.2	0.0
35	04/10/2022	GBPUSD	SELL	04/10/2022	05/10/2022	successful	0.01	114.4	0.0
36	05/10/2022	XAUUSD	SELL	05/10/2022	07/10/2022	successful	0.01	131.8	0.0
37	07/10/2022	XAUUSD	SELL	07/10/2022	10/10/2022	successful	0.01	105.7	0.0

38	10/10/2022	XAUUSD	SELL	10/10/2022	10/10/2022	successful	0.01	117.8	0.0
39	10/10/2022	GBPCAD	SELL	10/10/2022	10/10/2022	successful	0.01	65.7	0.0
40	10/10/2022	XAUUSD	SELL	10/10/2022	11/10/2022	successful	0.01	114.9	0.0
41	11/10/2022	XAUUSD	SELL	11/10/2022	13/10/2022	successful	0.01	104.4	0.0
42	13/10/2022	XAUUSD	SELL	13/10/2022	18/10/2022	unsuccessful	0.01	0.0	9.4
43	18/10/2022	GBPUSD	SELL	18/10/2022	19/10/2022	successful	0.01	65.2	0.0
44	18/10/2022	EURUSD	SELL	18/10/2022	19/10/2022	successful	0.01	40.1	0.0
45	19/10/2022	XAUUSD	SELL	19/10/2022	19/10/2022	successful	0.01	58.1	0.0
46	20/10/2022	XAUUSD	SELL	20/10/2022	21/10/2022	successful	0.02	23.0	0.0
47	21/10/2022	XAUUSD	SELL	21/10/2022	21/10/2022	successful	0.02	29.2	0.0
48	27/10/2022	XAUUSD	SELL	27/10/2022	27/10/2022	unsuccessful	0.01	0.0	25.3
49	27/10/2022	AUDJPY	SELL	27/10/2022	03/11/2022	successful	0.01	37.4	0.0
50	27/10/2022	XAUUSD	SELL	27/10/2022	28/10/2022	successful	0.01	46.9	0.0
51	01/11/2022	XAUUSD	SELL	01/11/2022	02/11/2022	unsuccessful	0.01	0.0	151.8
52	01/11/2022	AUDCAD	SELL	01/11/2022	03/11/2022	successful	0.01	30.4	0.0
53	03/11/2022	AUDJPY	SELL	03/11/2022	03/11/2022	successful	0.01	24.2	0.0
54	03/11/2022	EURJPY	SELL	03/11/2022	03/11/2022	successful	0.01	35.2	0.0
55	03/11/2022	XAUUSD	SELL	03/11/2022	03/11/2022	successful	0.01	27.4	0.0
56	04/11/2022	XAUUSD	SELL	04/11/2022	01/12/2022	unsuccessful	0.01	0.0	1494.1
57	15/11/2022	GBPNZD	SELL	15/11/2022	15/11/2022	successful	0.02	6.5	0.0
58	16/11/2022	GBPNZD	SELL	16/11/2022	16/11/2022	successful	0.01	11.0	0.0
59	16/11/2022	AUDJPY	SELL	16/11/2022	17/11/2022	successful	0.01	15.2	0.0
60	17/11/2022	USDCHF	BUY	17/11/2022	18/11/2022	successful	0.01	13.5	0.0
61	17/11/2022	AUDCAD	SELL	17/11/2022	18/11/2022	unsuccessful	0.01	0.0	42.5
62	22/11/2022	XAUUSD	BUY	22/11/2022	23/11/2022	successful	0.01	87.7	0.0
63	22/11/2022	GBPNZD	SELL	22/11/2022	23/11/2022	unsuccessful	0.01	0.0	23.5
64	24/11/2022	USDJPY	SELL	24/11/2022	24/11/2022	unsuccessful	0.01	0.0	21.7
65	24/11/2022	GBPJPY	SELL	24/11/2022	28/11/2022	successful	0.01	23.3	0.0
66	29/11/2022	EURAUD	SELL	29/11/2022	30/11/2022	successful	0.01	35.1	0.0
67	29/11/2022	USDCHF	BUY	29/11/2022	01/12/2022	unsuccessful	0.01	0.0	96.3
68	29/11/2022	CADJPY	SELL	29/11/2022	29/11/2022	successful	0.01	51.7	0.0
69	29/11/2022	XAUUSD	SELL	29/11/2022	01/12/2022	unsuccessful	0.01	0.0	300.2
70	06/12/2022	GBPCAD	BUY	06/12/2022	09/12/2022	successful	0.01	54.7	0.0
71	06/12/2022	CADJPY	SELL	06/12/2022	09/12/2022	successful	0.01	12.3	0.0
72	06/12/2022	EURCAD	BUY	06/12/2022	09/12/2022	successful	0.01	41.3	0.0
73	29/08/2022	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
74	02/09/2022	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

75	05/09/2022	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
76	12/09/2022	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
77	13/09/2022	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
78	14/09/2022	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
79	19/09/2022	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
80	22/09/2022	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
81	27/09/2022	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
82	28/09/2022	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
83	30/09/2022	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
84	03/10/2022	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
85	06/10/2022	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
86	12/10/2022	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
87	14/10/2022	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
88	17/10/2022	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
89	24/10/2022	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
90	25/10/2022	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
91	26/10/2022	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
92	28/10/2022	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
93	31/10/2022	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
94	02/11/2022	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
95	07/11/2022	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
96	08/11/2022	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
97	09/11/2022	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
98	10/11/2022	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
99	11/11/2022	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
100	14/11/2022	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
101	18/11/2022	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
102	21/11/2022	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
103	23/11/2022	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
104	25/11/2022	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
105	28/11/2022	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
106	30/11/2022	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
107	01/12/2022	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
108	02/12/2022	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
109	05/12/2022	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
110	07/12/2022	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
111	08/12/2022	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
112	09/12/2022	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

In [12]:
```
#make copy of dataset for cleaning
df_c = df.copy()
```

In [13]:
```
df_c.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 113 entries, 0 to 112
Data columns (total 14 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   Date                  113 non-null    object
 1   Pairs                 73 non-null     object
 2   Trade                 73 non-null     object
 3   open                  73 non-null     object
 4   close                 73 non-null     object
 5   result                73 non-null     object
 6   lot_size              73 non-null     float64
 7   no_of_pips_won        73 non-null     float64
 8   no_of_pips_lost       73 non-null     float64
 9   amt_won               73 non-null     float64
 10  amt_lost              73 non-null     float64
 11  Index_day             113 non-null    int64
 12  no_of_trades_entered  113 non-null    int64
 13  no_of_trades_closed   113 non-null    int64
dtypes: float64(5), int64(3), object(6)
memory usage: 13.2+ KB
```

In [14]:
```
#Fill null values in categorical columns with NO_TRADES
df_c[["Trade","result"]] =  df[["Trade","result"]].fillna('No_TRADES')
df_c.tail()
```

Out[14]:

	Date	Pairs	Trade	open	close	result	lot_size	no_of_pips_won	no_of_pips_lost	amt_won
108	02/12/2022	NaN	No_TRADES	NaN	NaN	No_TRADES	NaN	NaN	NaN	NaN
109	05/12/2022	NaN	No_TRADES	NaN	NaN	No_TRADES	NaN	NaN	NaN	NaN
110	07/12/2022	NaN	No_TRADES	NaN	NaN	No_TRADES	NaN	NaN	NaN	NaN
111	08/12/2022	NaN	No_TRADES	NaN	NaN	No_TRADES	NaN	NaN	NaN	NaN
112	09/12/2022	NaN	No_TRADES	NaN	NaN	No_TRADES	NaN	NaN	NaN	NaN

In [15]:
```
#convert index_day to categorical variable
df_c['Index_day'] = df_c['Index_day'].astype('category')
```

In [16]:
```
#Summary statistics of master dataset
df_c.describe()
```

Out[16]:

	lot_size	no_of_pips_won	no_of_pips_lost	amt_won	amt_lost	no_of_trades_entered	no_of_trades_close
count	73.000000	73.000000	73.000000	73.000000	73.000000	113.000000	113.00000
mean	0.010411	31.112329	37.013699	3.191644	3.701370	1.424779	1.14159
std	0.001999	33.001208	178.848065	3.312133	17.884807	1.266359	1.20909
min	0.010000	0.000000	0.000000	0.000000	0.000000	0.000000	0.00000
25%	0.010000	0.000000	0.000000	0.000000	0.000000	0.000000	0.00000
50%	0.010000	23.300000	0.000000	2.420000	0.000000	1.000000	1.00000
75%	0.010000	41.600000	3.600000	4.600000	0.360000	2.000000	2.00000

max	0.020000	131.800000	1494.100000	13.180000	149.410000	4.000000	5.00000

In [17]:
```python
#summary statistics of second dataset
df2.describe()
```

Out[17]:

	Index_day	no_of_trades_entered	no_of_trades_closed
count	80.0000	80.000000	80.000000
mean	40.5000	0.912500	0.912500
std	23.2379	1.093059	1.081417
min	1.0000	0.000000	0.000000
25%	20.7500	0.000000	0.000000
50%	40.5000	0.500000	1.000000
75%	60.2500	2.000000	1.000000
max	80.0000	4.000000	5.000000

In [18]:
```python
#convert date, open and close columns to datetime
df_c[['Date', 'open','close']] = df_c[['Date', 'open','close']].apply(pd.to_datetime, da
```

In [19]:
```python
df_c.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 113 entries, 0 to 112
Data columns (total 14 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   Date                  113 non-null    datetime64[ns]
 1   Pairs                 73 non-null     object
 2   Trade                 113 non-null    object
 3   open                  73 non-null     datetime64[ns]
 4   close                 73 non-null     datetime64[ns]
 5   result                113 non-null    object
 6   lot_size              73 non-null     float64
 7   no_of_pips_won        73 non-null     float64
 8   no_of_pips_lost       73 non-null     float64
 9   amt_won               73 non-null     float64
 10  amt_lost              73 non-null     float64
 11  Index_day             113 non-null    category
 12  no_of_trades_entered  113 non-null    int64
 13  no_of_trades_closed   113 non-null    int64
dtypes: category(1), datetime64[ns](3), float64(5), int64(2), object(3)
memory usage: 15.1+ KB
```

In [20]:
```python
#generate new column called trade length to indicate the duration of each trade
df_c['trade_length']= (df_c['close'] - df_c['open']).dt.days
df_c.head()
```

Out[20]:

	Date	Pairs	Trade	open	close	result	lot_size	no_of_pips_won	no_of_pips_lost	amt_won	amt_los
0	2022-08-22	GBPJPY	SELL	2022-08-22	2022-08-23	successful	0.01	34.2	0.0	3.42	0.0
1	2022-08-22	AUDUSD	SELL	2022-08-22	2022-08-22	successful	0.01	16.8	0.0	1.68	0.0
2	2022-08-23	GBPCHF	BUY	2022-08-23	2022-08-23	successful	0.01	33.2	0.0	3.32	0.0
3	2022-	AUDUSD	SELL	2022-	2022-	unsuccessful	0.01	0.0	39.0	0.00	3.9

	08-24			08-24	08-25						
4	2022-08-25	GBPJPY	SELL	2022-08-25	2022-08-26	unsuccessful	0.01	0.0	15.9	0.00	1.5

ANALYSIS AND VISUALIZATION

In [144...

```
#Set colour palette and style
sns.set_style('darkgrid')
sns.set(rc={"figure.figsize": (12, 8)})
font_labels = 15
font_title = 18
```

Defining functions

In [22]:

```
def Cntpltx(x, title, xlabel):
    '''This function plots vertical count graphs
    '''
    #arrange the bars in order of frequency
    count_a = x.value_counts()[:15]
    count_b = x.value_counts(normalize = True)[:15]*100

    #plot the count graph
    palette= ['gray'] * 20
    palette[0] = 'forestgreen'
    ax = sns.countplot(x = x, order = x.value_counts()[:15].index, palette = palette)

    #create the labels
    label = [f' {p[0]} | {p[1]:.2f}%' for p in zip(count_a, count_b)]
    ax.bar_label(container=ax.containers[0], labels=label)

    #graph labels
    plt.title(title, fontsize = font_title)
    plt.xlabel(xlabel, fontsize = font_labels)
    plt.ylabel('')
    plt.yticks([])
```

In [23]:

```
def Cntplty(y, title, ylabel):
    '''This function plots horizontal count graphs
    '''
    #arrange the bars in order of frequency
    count_a = y.value_counts()[:15]
    count_b = y.value_counts(normalize = True)[:15]*100

    #plot the count graph
    palette= ['gray'] * 20
    palette[0] = 'forestgreen'
    ax = sns.countplot(y = y, order = y.value_counts()[:20].index, palette = palette)


    #create the labels
    label = [f' {p[0]} | {p[1]:.2f}%' for p in zip(count_a, count_b)]
    ax.bar_label(container=ax.containers[0], labels=label)



    #graph labels
    plt.title(title, fontsize = font_title)
    plt.ylabel(ylabel, fontsize = font_labels)
```

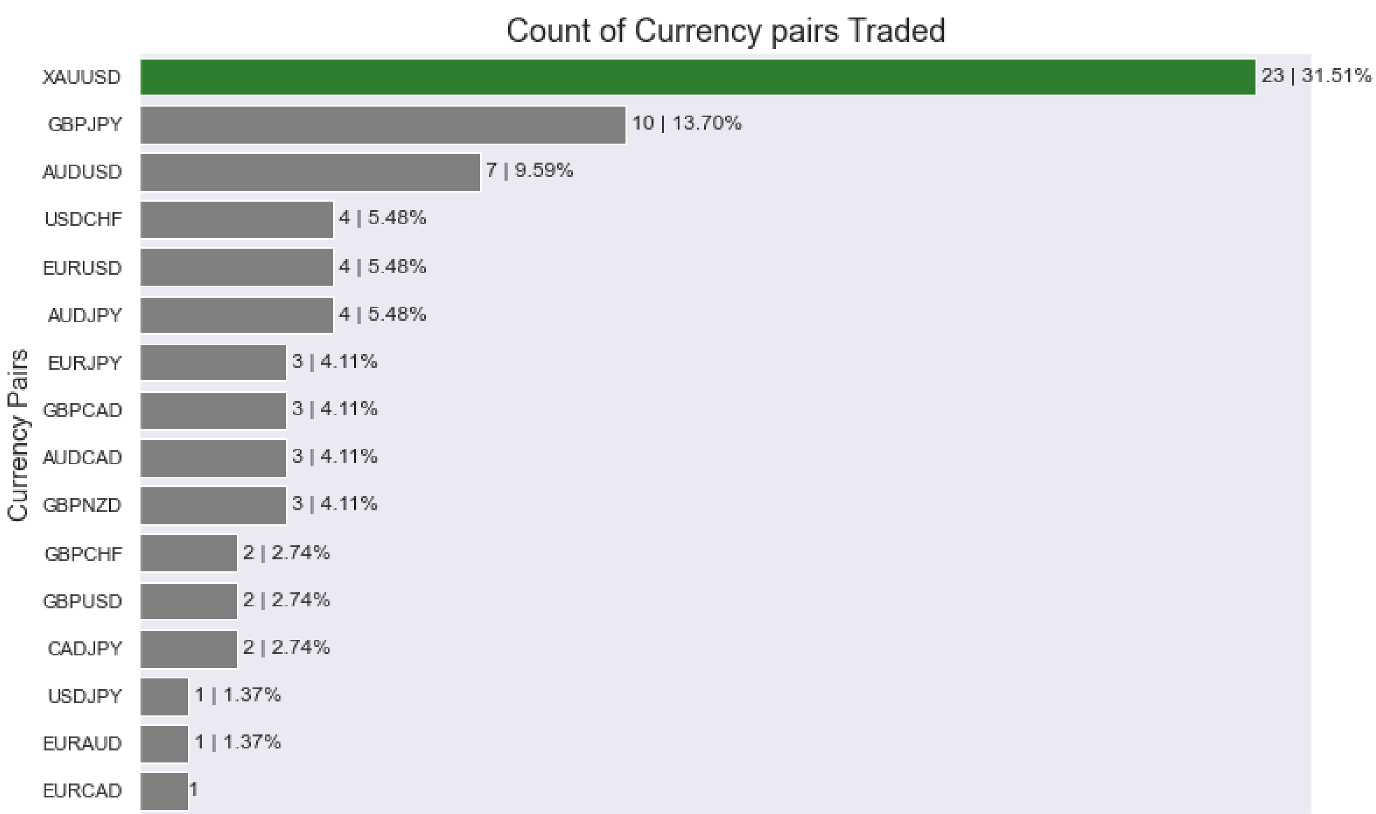
```
    plt.xlabel('')
    plt.xticks([])
```

In [80]:
```
#create function to plot scatterplot
def plot_scatter1(x, y, title, xlabel, ylabel, transparency):
    '''This function plots a single scatterplot'''

    #regression plot
    sns.regplot(data = df_c, x=x, y=y, x_jitter=0.3, y_jitter=0.3, scatter_kws={'alpha':


    #display graph labels
    plt.xlabel(xlabel, fontsize=16)
    plt.ylabel(ylabel, fontsize=16)
    plt.title(title, fontsize=22)
```

In [165…
```
#creating a function to plot barcharts
def plot_bar2(subplot ,x , title, xlabel, ylabel):
    '''This function plots multiple barcharts'''

    #defining subplot locations
    ax = plt.subplot(1 ,2 ,subplot)

    #plot the barchart
    sns.countplot(data=df_c, x= x, color =colors)

    #display graph labels
    plt.xlabel(xlabel, fontsize=16)
    plt.ylabel(ylabel, fontsize=16)
    plt.title(title, fontsize=22)
```

What currency pair was most traded?

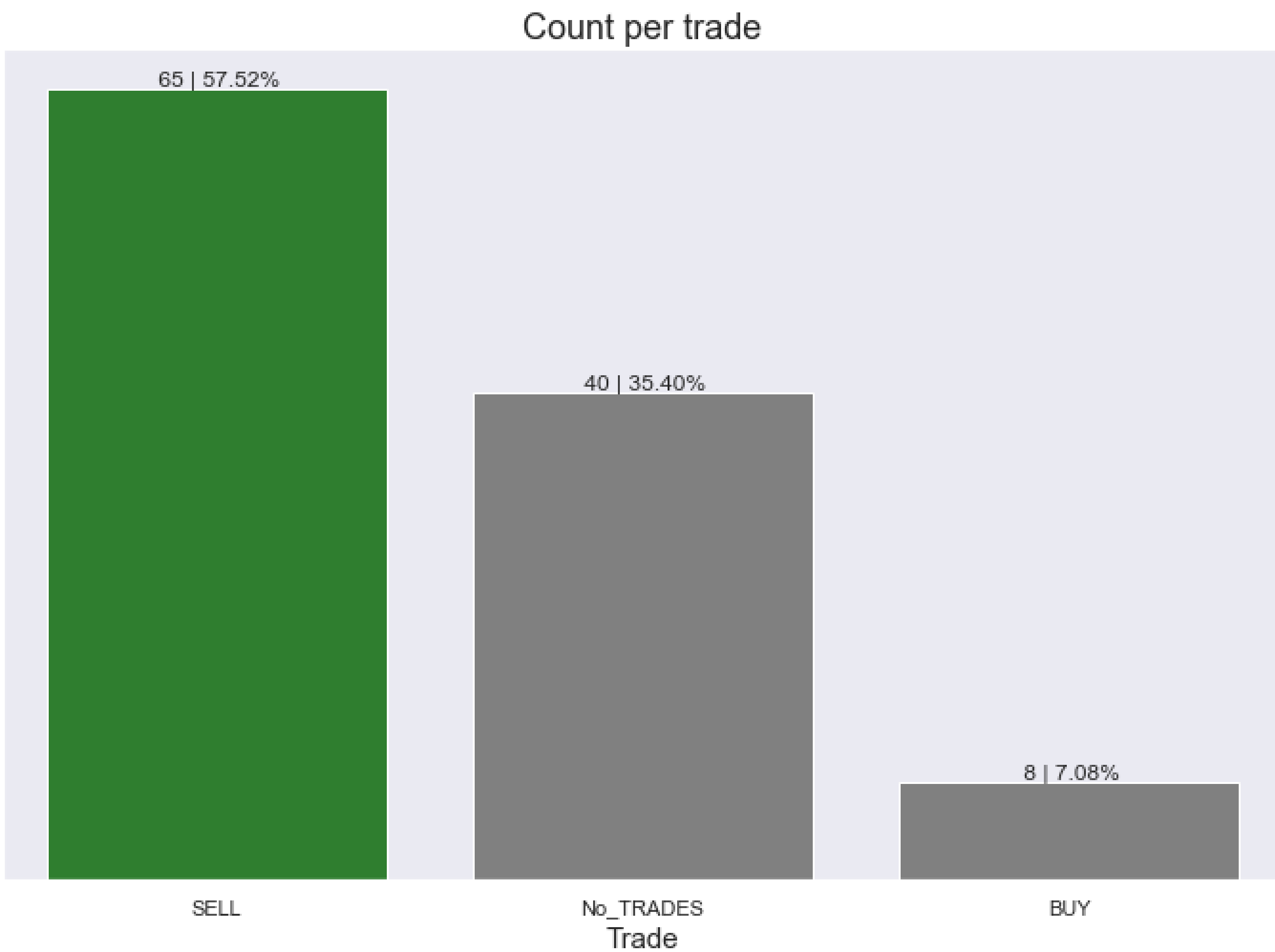
In [25]:
```
#Plot barchat
Cntplty(df_c['Pairs'], "Count of Currency pairs Traded", "Currency Pairs")
```



The XAUUSD currency pair was most traded during the 80-days trade period. It accounted for 31.5% of all trades or 23 trades. The GBPJPY pair was next at 13.7% or 10 trades. This is over 50 percent less that the most traded pair.

What trades were most entered: Buys or sells?

In [68]:
```
#Plot barchat
Cntpltx(df_c['Trade'], 'Count per trade', 'Trade')
```



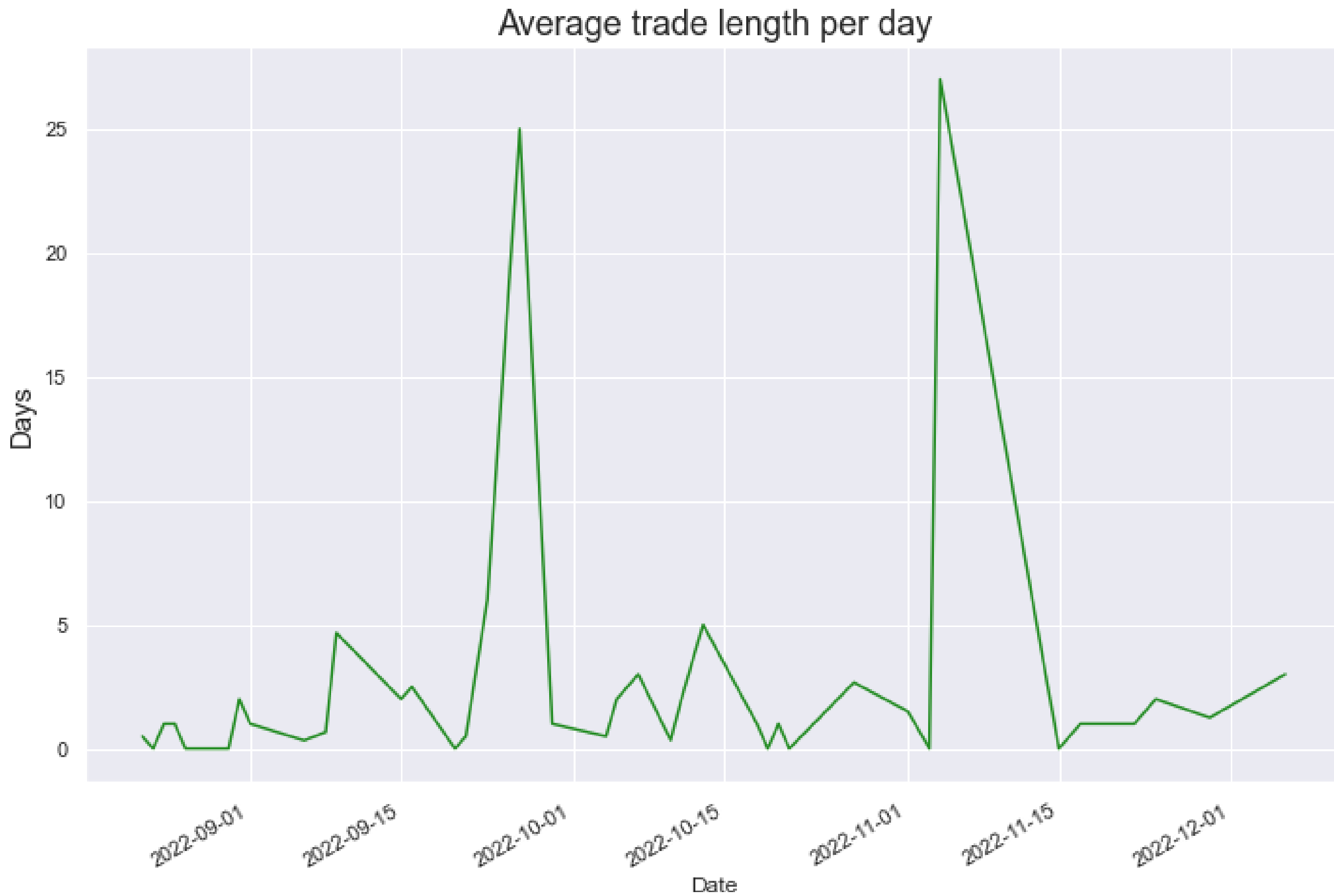
Over 57% of trades called were sell trades. About 35% of the time, trades were not called. Buy trades were called only 7% of the whole time.

What was the average trade length per day?

In [27]:
```
#get mean of trade length column
df_c['trade_length'].mean()
```

Out[27]:
```
2.0547945205479454
```

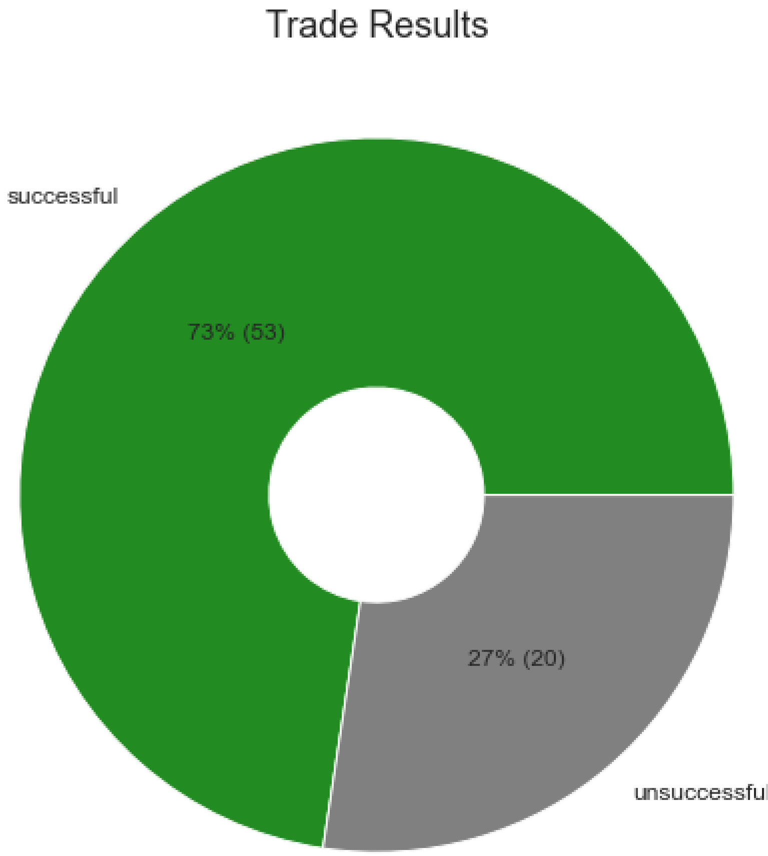
In [146…
```
#Group by date and plot mean per day
avg_trade_length = df_c.groupby('Date')['trade_length'].agg('mean')
avg_trade_length.dropna().plot(color='forestgreen')
plt.title('Average trade length per day', fontsize = font_title)
plt.ylabel('Days', fontsize = font_labels)
plt.show()
```



Generally, an average trade length of 2 days was observed during the 80 days trade period. Upon further analysis, i observed that although the 'in-a-trade' duration ranged between 0 and 5, two spikes showed two trades which were drawn out for about 25-30days.

What percentage of trades were successful?

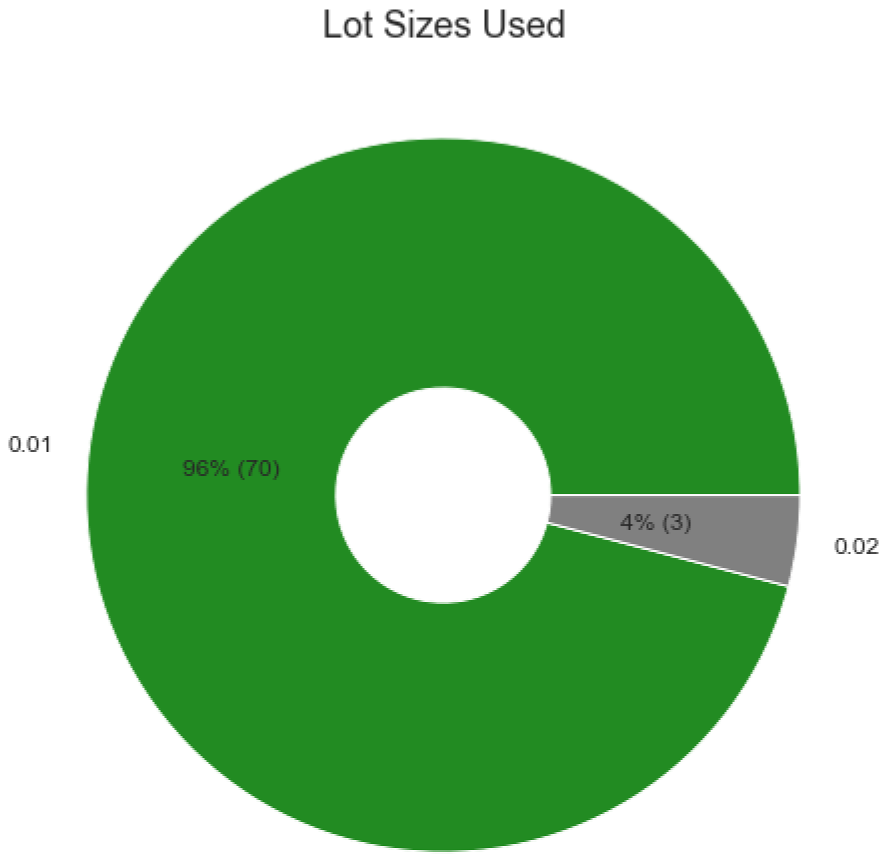
In [29]:
```
#create donut chart
plt.figure(figsize=[8,12])
colors = ['forestgreen', 'gray']
a = df['result'].value_counts()
a.plot(kind="pie",  autopct=lambda p: '{:.0f}% ({:.0f})'.format(p,(p/100)*a.sum()),
        textprops={'fontsize': 12}, wedgeprops = {'width' : 0.7}, colors = colors)

#display graph labels
plt.title('Trade Results', fontsize = font_title)
plt.ylabel('')
plt.show();
```



73% of trades called were successful, while 27% of trades were unsuccessful. This shows an accuracy of about 73% on closed trades.

What lot size was most used?

In [30]:
```
#create donut chart
plt.figure(figsize=[8,12])
colors = ['forestgreen', 'gray']
a = df['lot_size'].value_counts()
a.plot(kind="pie",  autopct=lambda p: '{:.0f}% ({:.0f})'.format(p,(p/100)*a.sum()),
        textprops={'fontsize': 12}, wedgeprops = {'width' : 0.7}, colors = colors)

#display graph labels
plt.title('Lot Sizes Used', fontsize = font_title)
plt.ylabel('')
plt.show();
```



96% of the time, the 0.01 lot size was used. This was because the model account was 200 dollar account.

What distribution of pip value was observed ?

In [31]:
```
#Round up StatedMonthlyIncome column
df_c['no_of_pips_won'] = df_c['no_of_pips_won'].round()
```

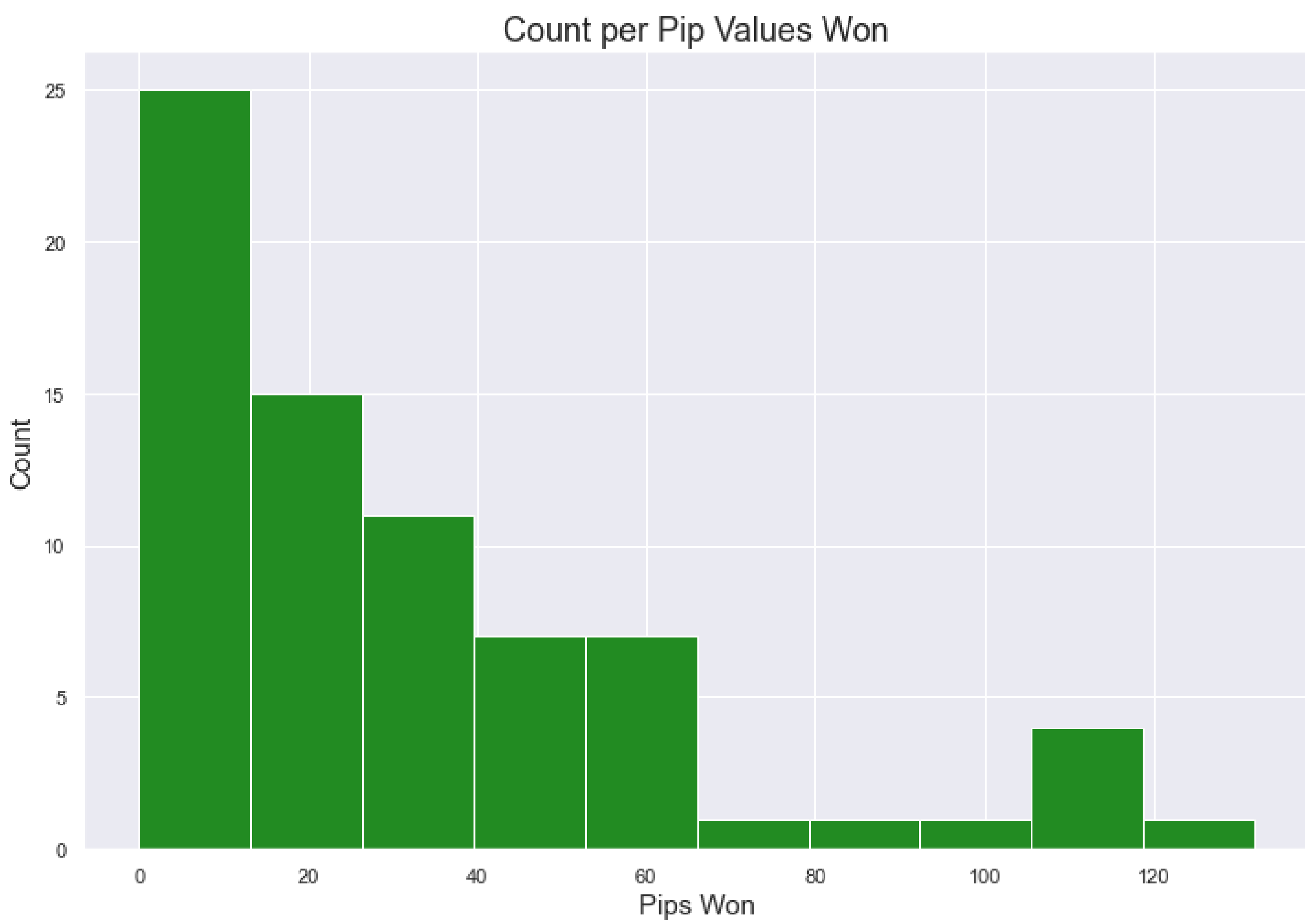
In [36]:
```
df_c['no_of_pips_won'].value_counts()
```

Out[36]:
```
0.0      20
23.0      3
35.0      2
32.0      2
14.0      2
40.0      2
64.0      2
24.0      2
47.0      1
104.0     1
65.0      1
58.0      1
29.0      1
37.0      1
34.0      1
30.0      1
66.0      1
27.0      1
6.0       1
11.0      1
15.0      1
88.0      1
52.0      1
```

```
55.0      1
12.0      1
115.0     1
114.0     1
118.0     1
31.0      1
33.0      1
20.0      1
16.0      1
9.0       1
22.0      1
19.0      1
18.0      1
4.0       1
106.0     1
45.0      1
26.0      1
60.0      1
42.0      1
57.0      1
17.0      1
132.0     1
41.0      1
Name: no_of_pips_won, dtype: int64
```

In [147…

```
#Plot histogram
plt.hist(df_c['no_of_pips_won'], bins=10, color='forestgreen')
plt.title('Count per Pip Values Won', fontsize = font_title)
plt.ylabel('Count', fontsize = font_labels)
plt.xlabel('Pips Won', fontsize = font_labels)
plt.show()
```

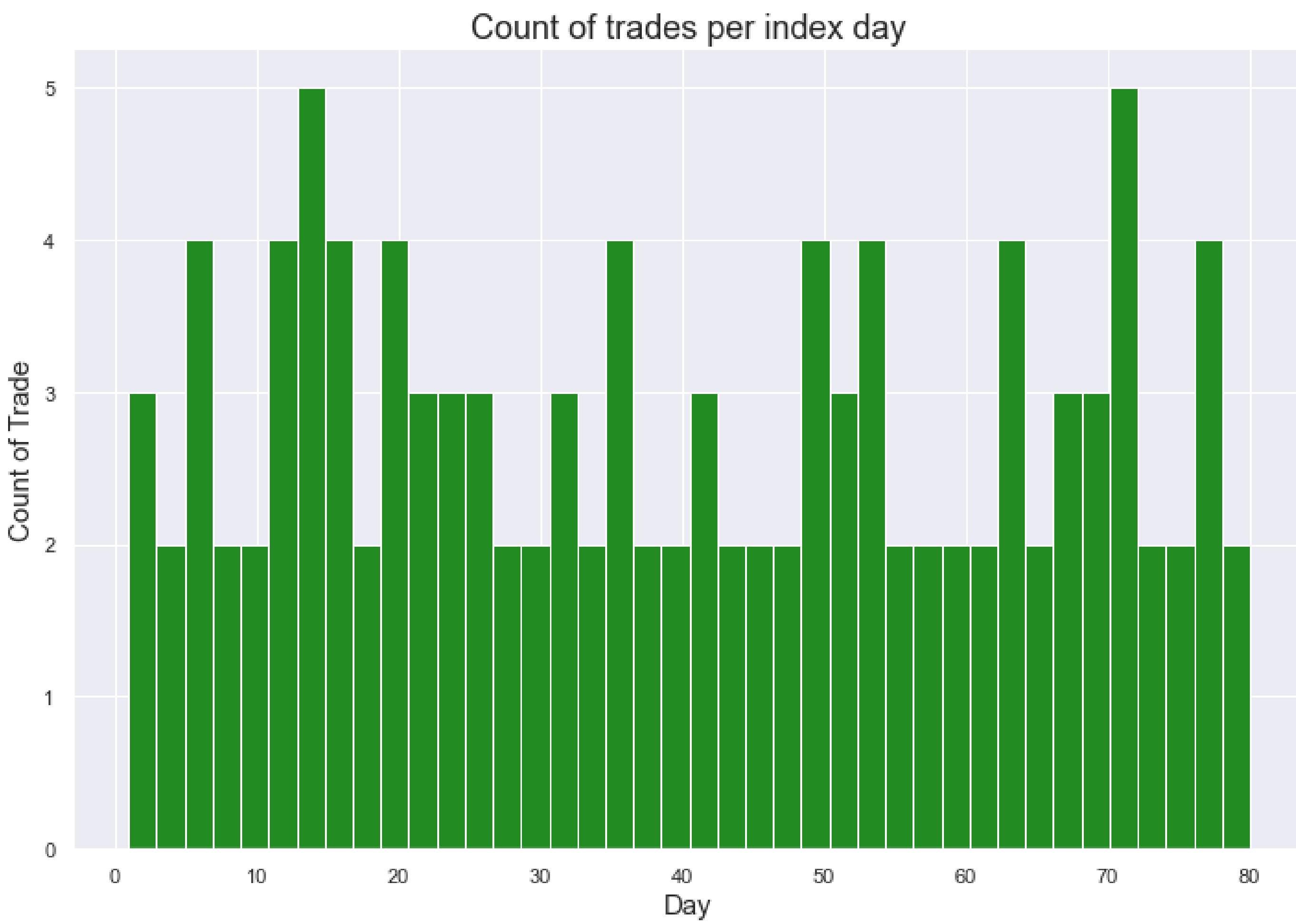


This histogram is right skewed, infering that more often than not, pip values of less than 70 were observed, compared to higher values above 70. A spike is also observed at the 110 pip value. This indicates a hiigher count at that level, compared to other larger values.

What number of trades were observed throughout the index days?

In [148...

```
#plot histogram showng destribution of trades across each index day
plt.hist(df_c['Index_day'], bins=40, color='forestgreen')
plt.title('Count of trades per index day', fontsize = font_title)
plt.ylabel('Count of Trade', fontsize = font_labels)
plt.xlabel('Day', fontsize = font_labels)
plt.show()
```



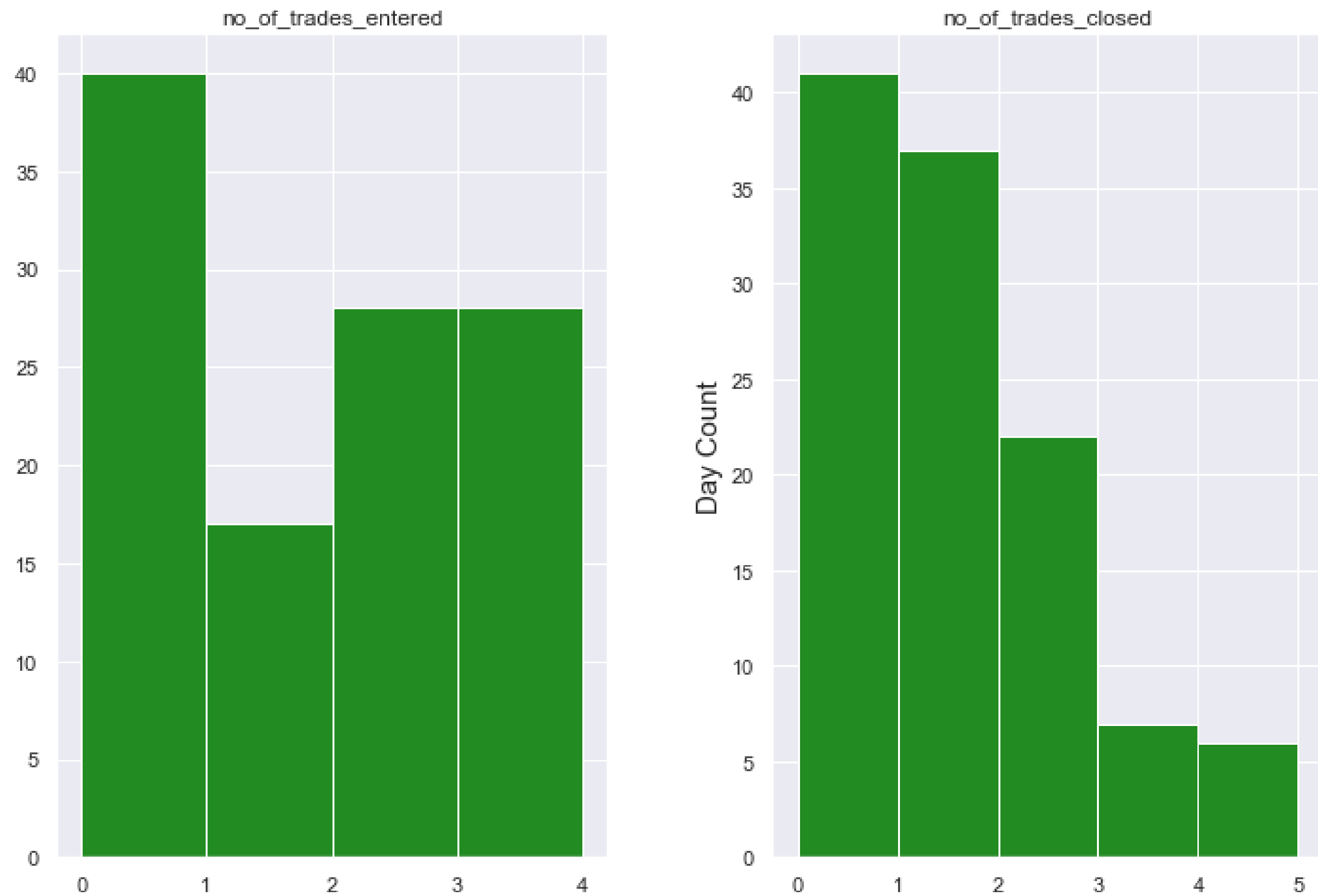
A multimodal histogram is observed. Ths shows the maximum trades entered n a day was 5. More frequent was the 4 trade count in a day. The 2 trades a day level also had a significant number of mode values.

Is there any similarity or difference between the number of trades entered and closed?

In [163...

```
#defining subplot locations
fig, axes = plt.subplots(1, 2)

#plot the histograms
df_c.hist('no_of_trades_entered', bins=4, color='forestgreen', ax=axes[0])
df_c.hist('no_of_trades_closed', bins=5, color='forestgreen', ax=axes[1])


plt.ylabel('Day Count', fontsize = font_labels)
```

```
plt.show()
```



We can observe higher chances of opening 3 and 4 trades in a day, compared to closing above 3 trades in a day. A decline is observed when closing from 3 trades and above.

What month had the highest opened trades?

In [116…

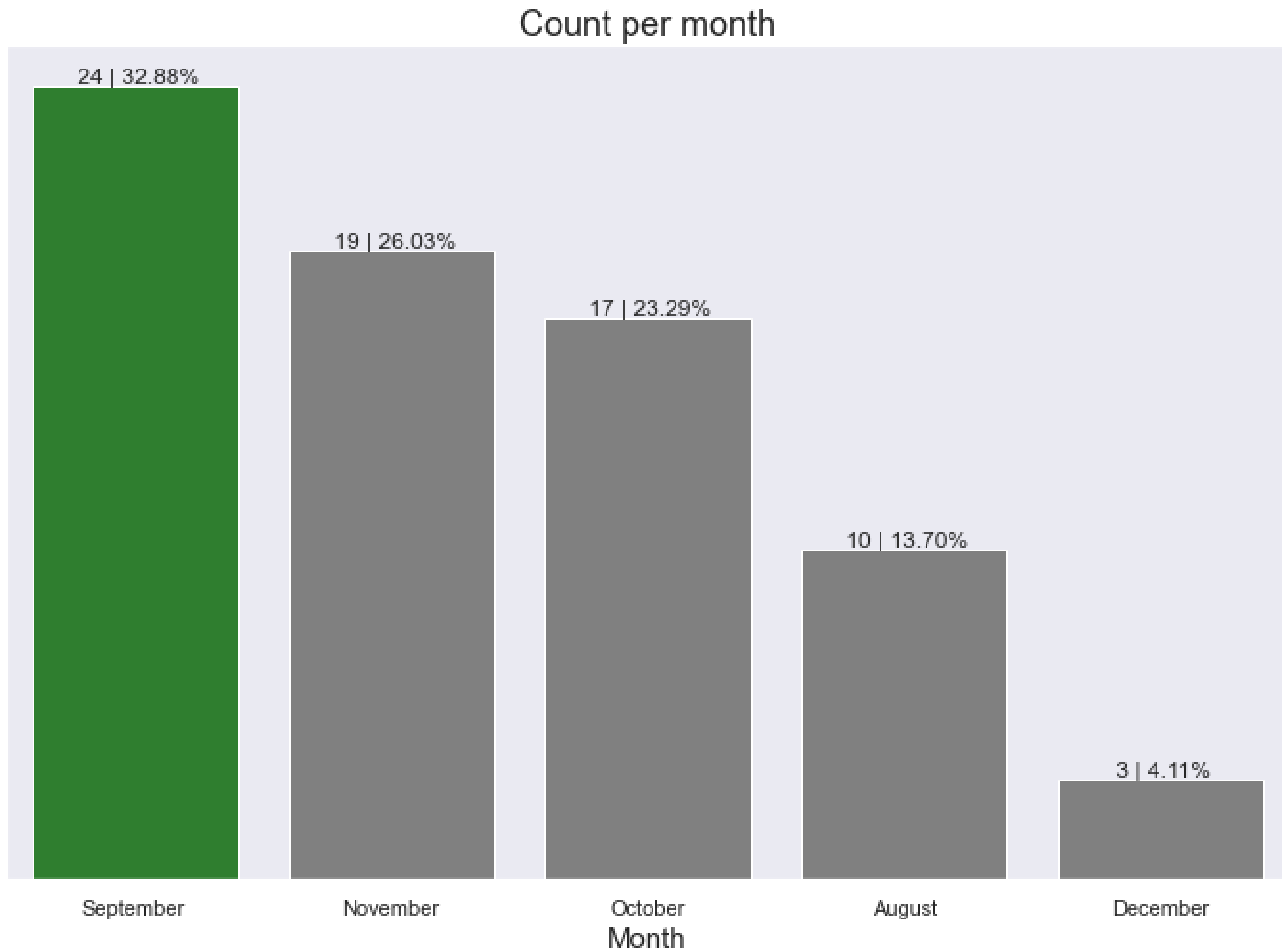
```
#Create new columns for the year, month and weekday of loan origination

df_c['open_month'] = df_c['Date'].dt.month_name()
df_c['close_month'] = df_c['close'].dt.month_name()
df_c['open_day'] = df_c['Date'].dt.day_name()
df_c['close_day'] = df_c['close'].dt.day_name()
df_c.head(3)
```

Out[116]:

	Date	Pairs	Trade	open	close	result	lot_size	no_of_pips_won	no_of_pips_lost	amt_won	amt_lost
0	2022-08-22	GBPJPY	SELL	2022-08-22	2022-08-23	successful	0.01	34.0	0.0	3.42	0.0
1	2022-08-22	AUDUSD	SELL	2022-08-22	2022-08-22	successful	0.01	17.0	0.0	1.68	0.0
2	2022-08-23	GBPCHF	BUY	2022-08-23	2022-08-23	successful	0.01	33.0	0.0	3.32	0.0

In [72]:

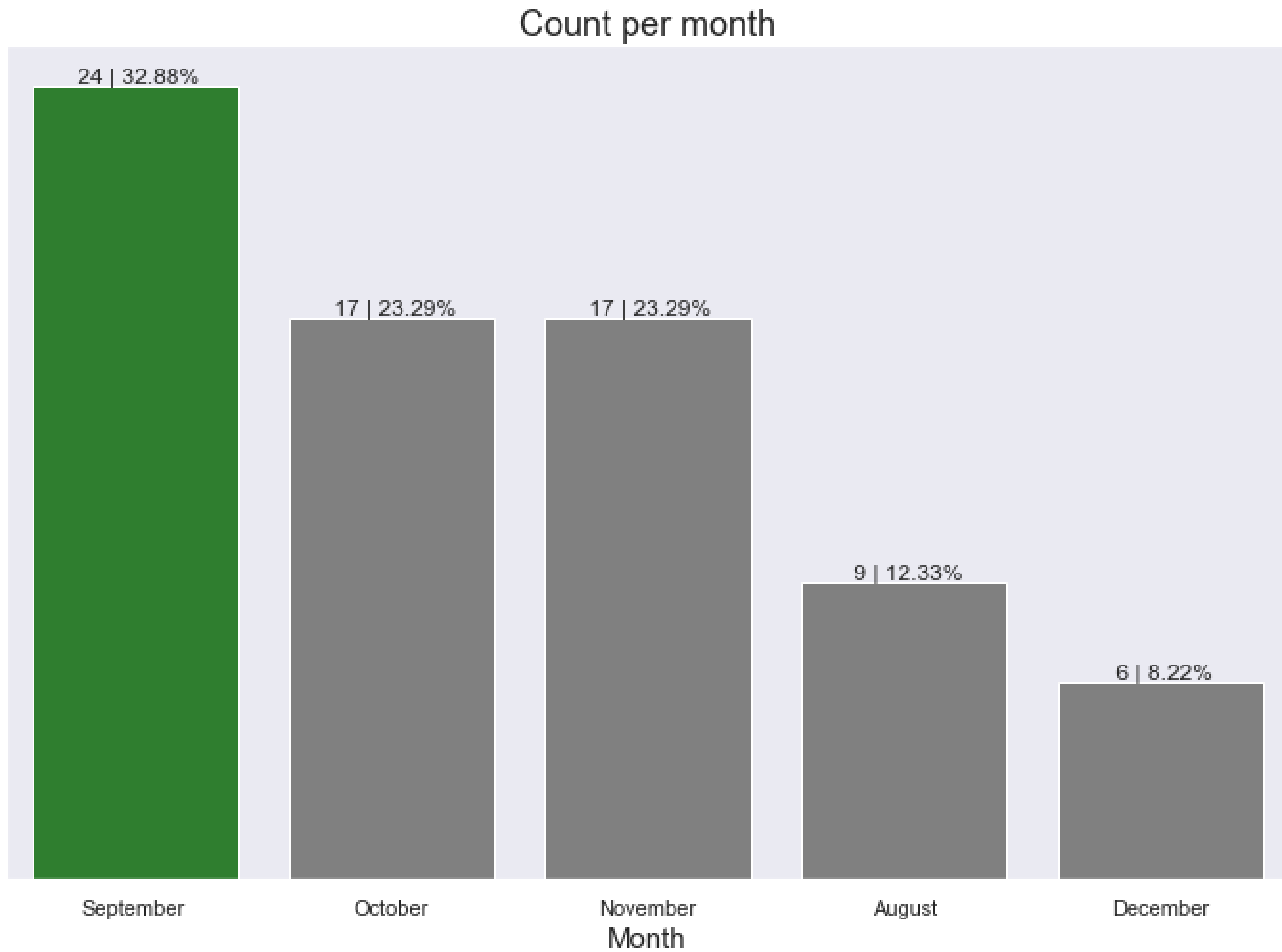
```
#no of opened trades per month
Cntpltx(df_c['open_month'], 'Count per month', 'Month')
```



September recorded the highest number of opened trades

What month had the highest closed trades?

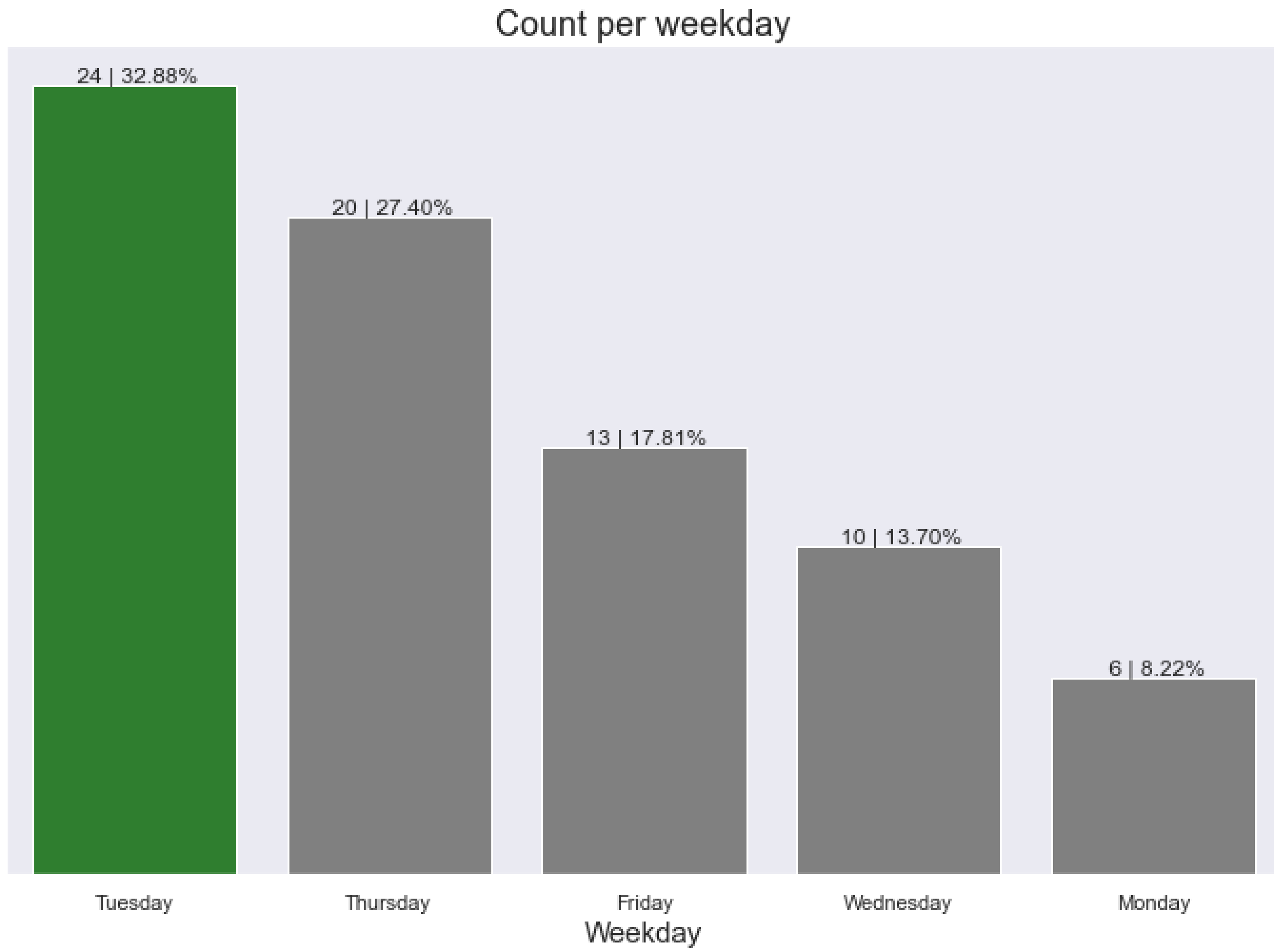
In [73]:
```
#no of closed trades per month
Cntpltx(df_c['close_month'], 'Count per month', 'Month')
```



As expected, most opened trades had an average trade length of 0-5 days. This is why september also recorded the highest number of closed trades as well.

What weekday had the highest trades?

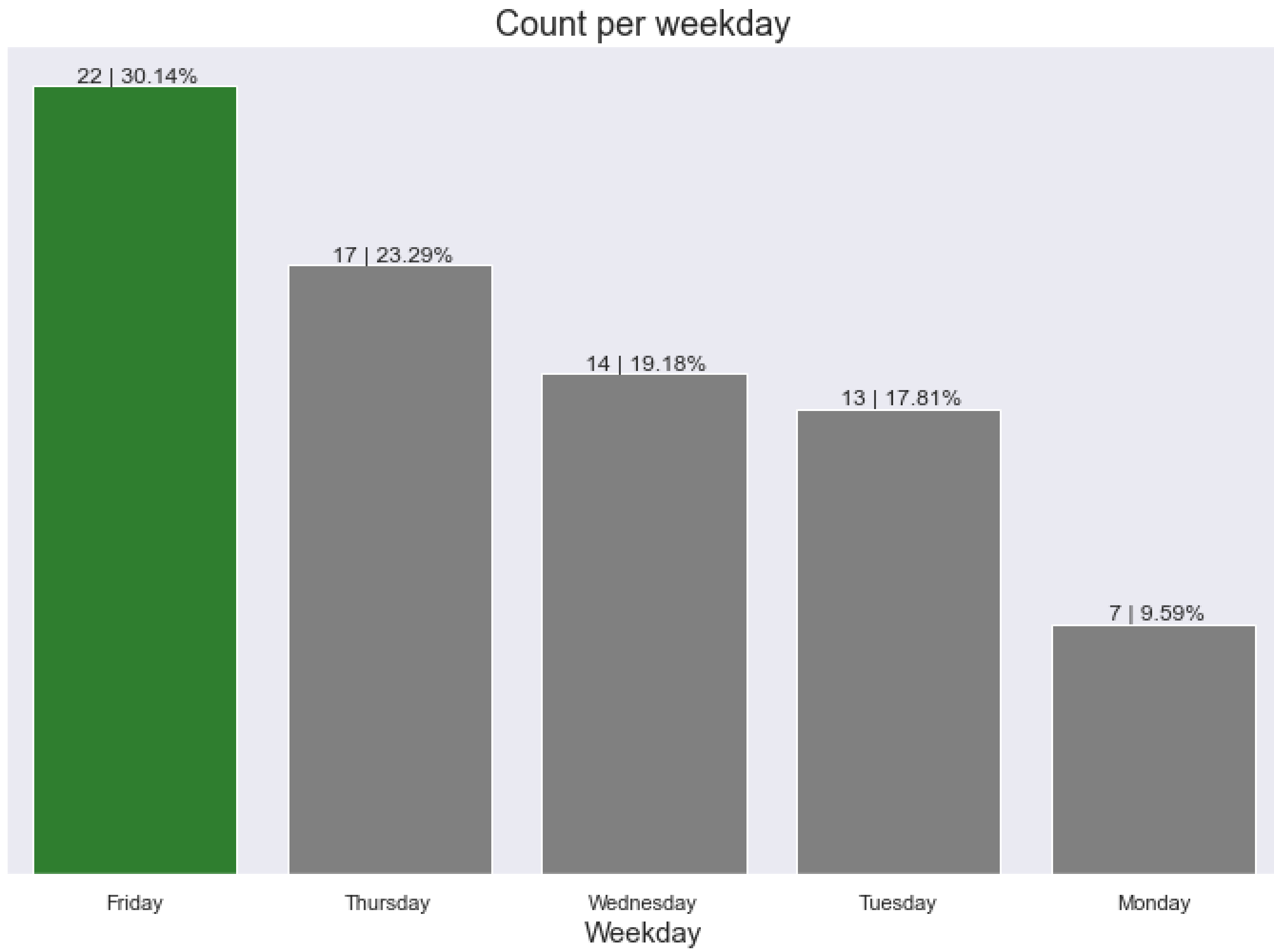
In [74]:
```
#no of opened trades per weekday
Cntpltx(df_c['open_day'], 'Count per weekday', 'Weekday')
```



About 32% of all trades were opened on tuesday. This was the highest, compared to other weekdays. This is due to the fact that mondays were usually used to study the trends of the market for the week.

In [75]:

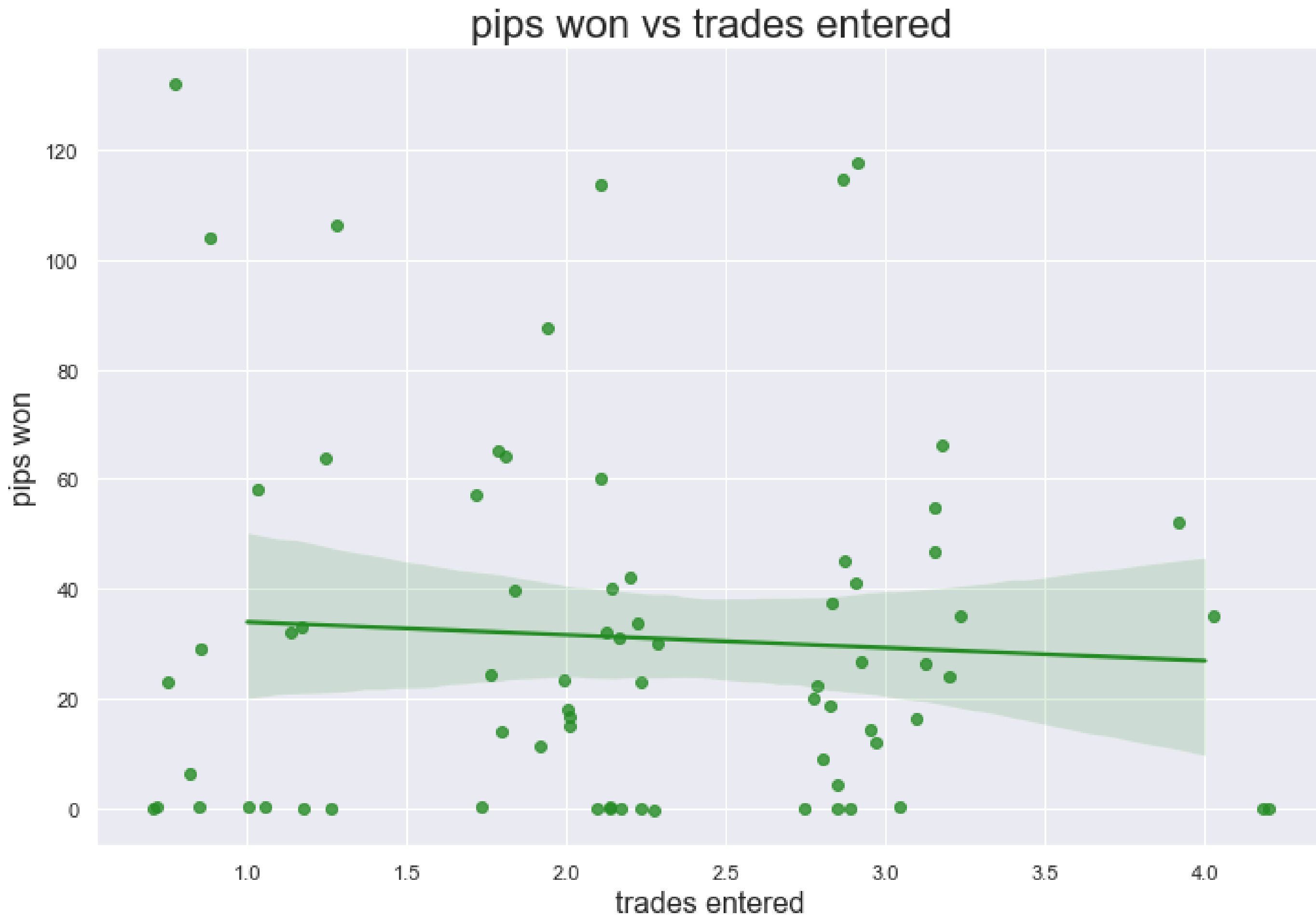
```
#no of closed trades per weekday
Cntpltx(df_c['close_day'], 'Count per weekday', 'Weekday')
```



Most trades were closed on fridays, which marks the end of the trading week. There is no currency pair trading during the weekend. Trading usually resumes on sunday evening.

Is there any relationship between the number of pips won and the number of trades entered?

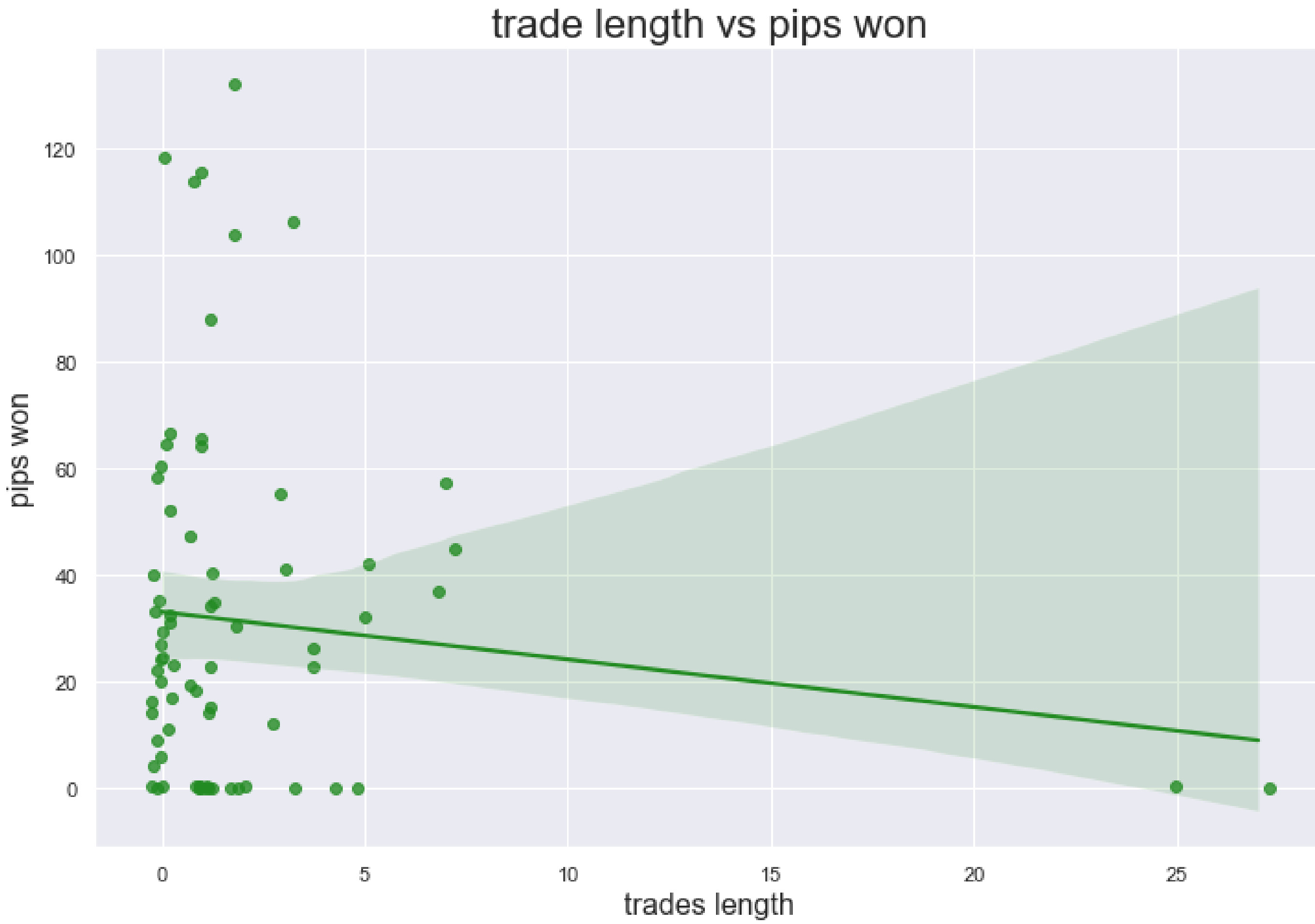
In [90]:
```
#plot scatter plot
plot_scatter1('no_of_trades_entered', 'no_of_pips_won', 'pips won vs trades entered','tr
```



No significant relationship was observed between the number of pips won and the number of trades entered

Is there any relationship between the trade length and the number of pips won?

In [92]:
```
plot_scatter1('trade_length','no_of_pips_won',  'trade length vs pips won','trades lengt
#plt.xlim (-0.5, 8)
```



A negative correlation was observed between the trade length and the number of pips won. This means, the longer the trade length, the lower the number of pips to be won. Longer trade days often led to lower pips, and losses in some cases.

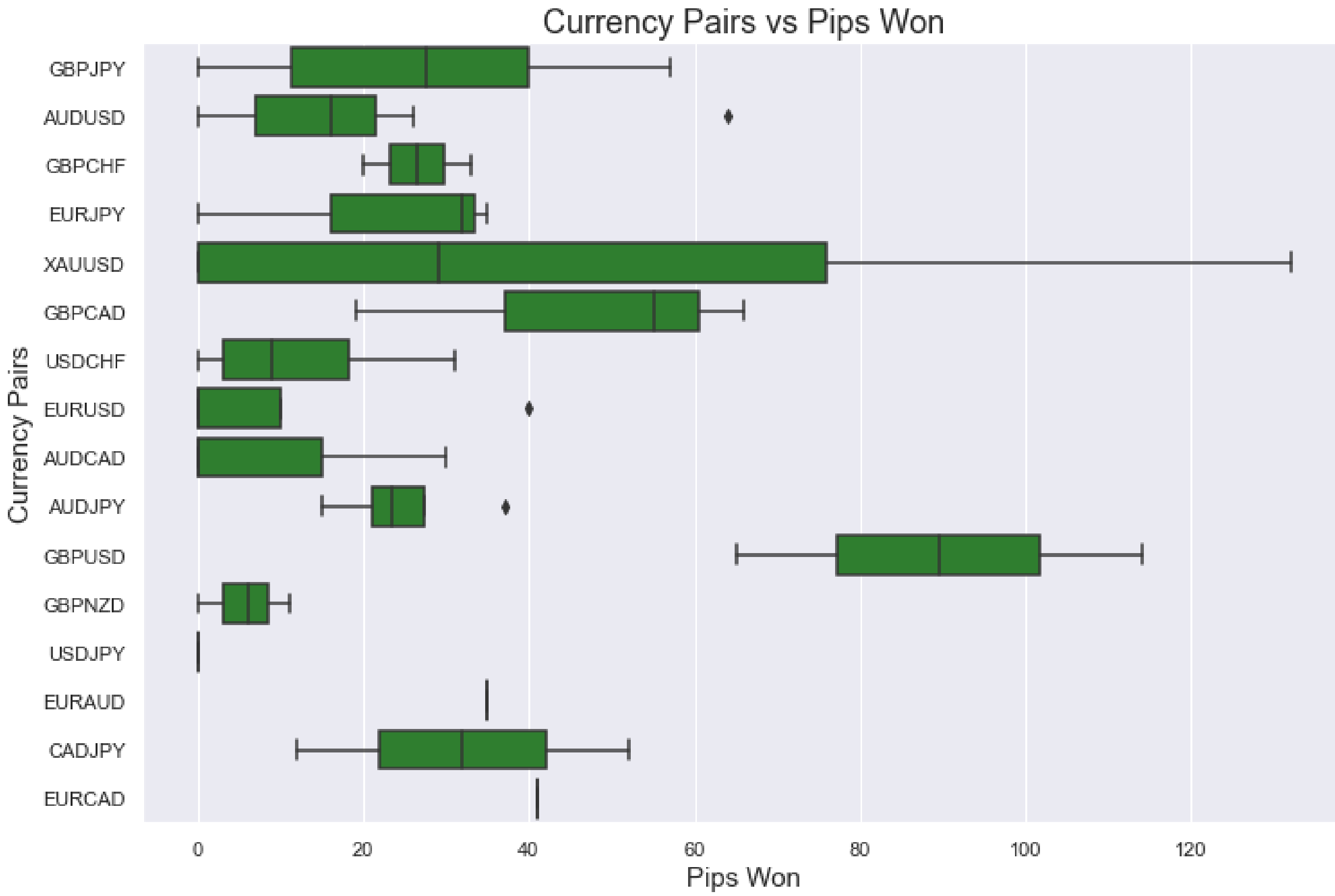
What currency pair gave the highest number of pips won?

In [181…

```
#Boxplot plot comparing Prosper rating and loan amount
sns.boxplot(data = df_c, y='Pairs', x= 'no_of_pips_won', color= 'forestgreen')
plt.title('Currency Pairs vs Pips Won', fontsize = font_title)
plt.ylabel('Currency Pairs', fontsize = font_labels)
plt.xlabel('Pips Won', fontsize = font_labels);
```

Out[181]:

```
Text(0.5, 0, 'Pips Won')
```



XAUUSD won the highest number of pips overall, however, the top 75% of all GBPUSD trades gave higher pips than the lower 75% of all XAUUSD trades. The median GBPUSD trade is higher than the median XAUUSD trade, but the top 25% of XAUUSD trades puts it in the lead. USDJPY recorded the least number of pips won overall.

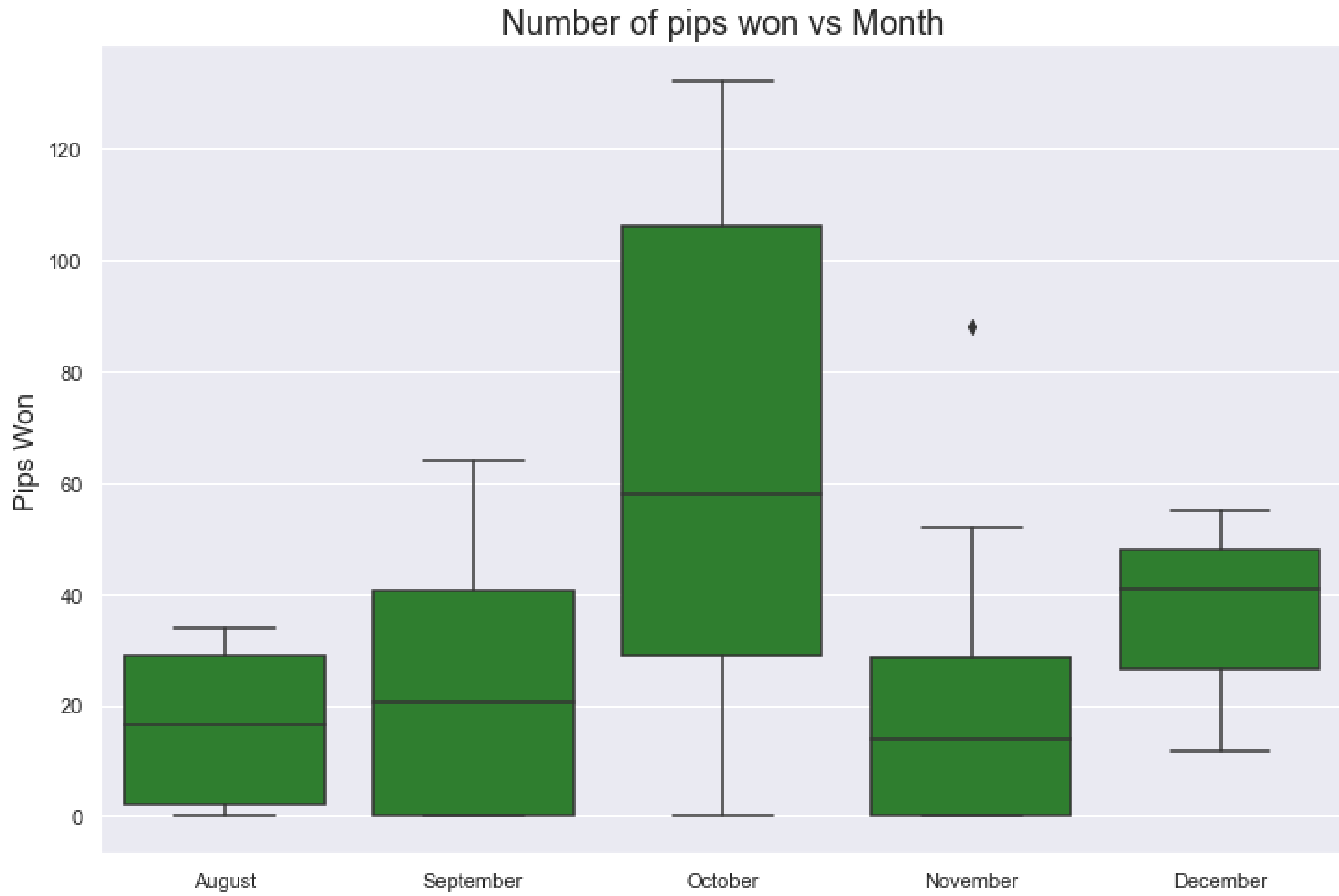
What month recorded the highest number of pips won?

In [177…
```
#Find the total number of pips won
df['no_of_pips_won'].sum()
```

Out[177]:
```
2271.2
```

In [172…
```
#Boxplot plot comparing month and number of pips won
sns.boxplot(data = df_c, x='open_month', y= 'no_of_pips_won', color= 'forestgreen')
plt.title('Number of pips won vs Month', fontsize = font_title)
plt.ylabel('Pips Won', fontsize = font_labels)
plt.xlabel(' ', fontsize = font_labels);
```

Out[172]:
```
Text(0.5, 0, ' ')
```

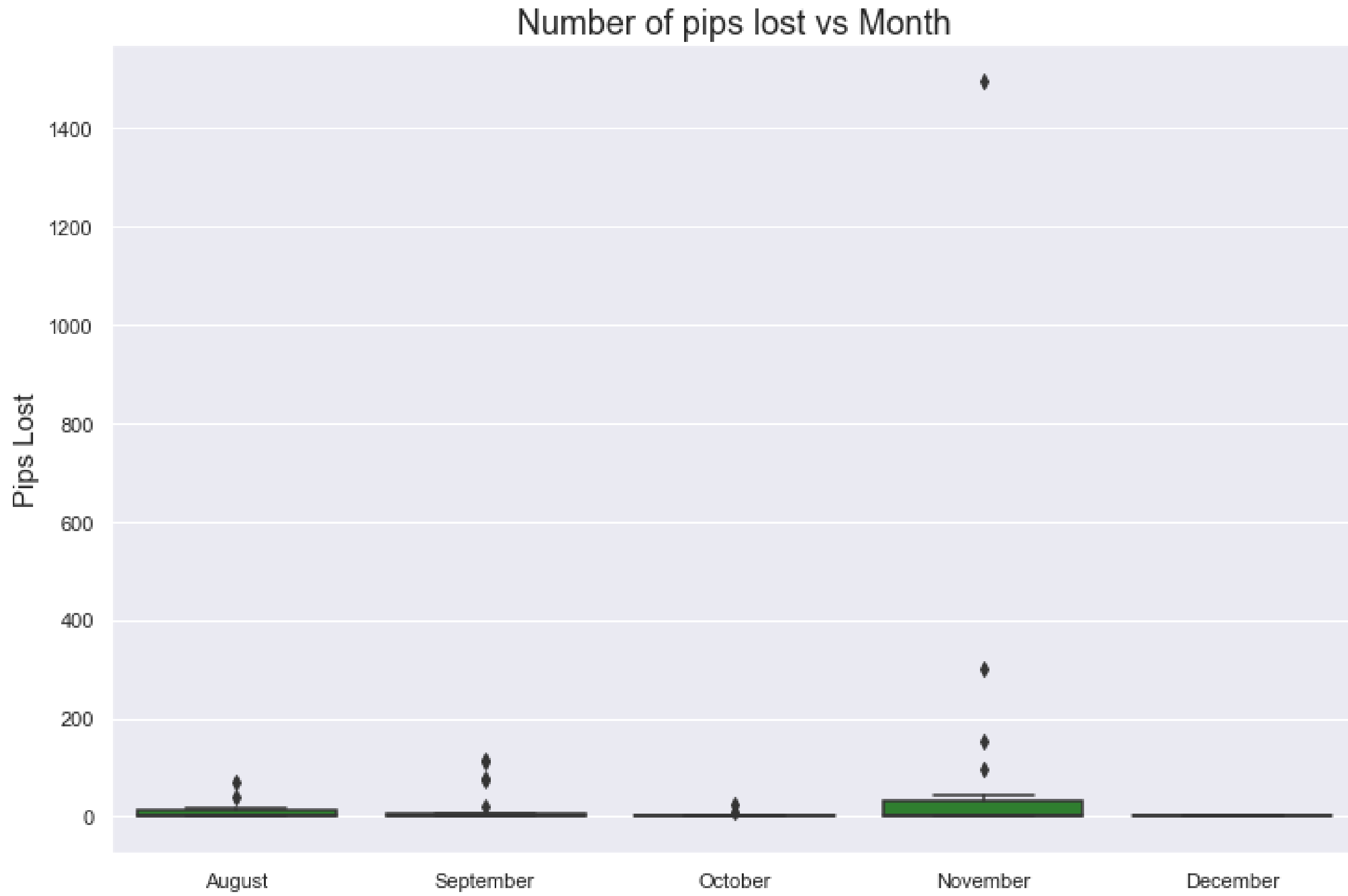


October recorded the highest number of pips won. **2271.2** pips were won throughout the 80 day trade period

What month recorded the highest number of pips lost?

In [178…
```
#Find the total number of pips lost
df['no_of_pips_lost'].sum()
```
Out[178]:
```
2701.9999999999995
```

In [173…
```
#Boxplot plot comparing month and number of pips lost
sns.boxplot(data = df_c, x='open_month', y= 'no_of_pips_lost', color= 'forestgreen')
plt.title('Number of pips lost vs Month', fontsize = font_title)
plt.ylabel('Pips Lost', fontsize = font_labels)
plt.xlabel(' ', fontsize = font_labels);
```
Out[173]:
```
Text(0.5, 0, ' ')
```



November recorded the highest number of pips lost. **2702** pips were lost throughout the 80 day trade period.

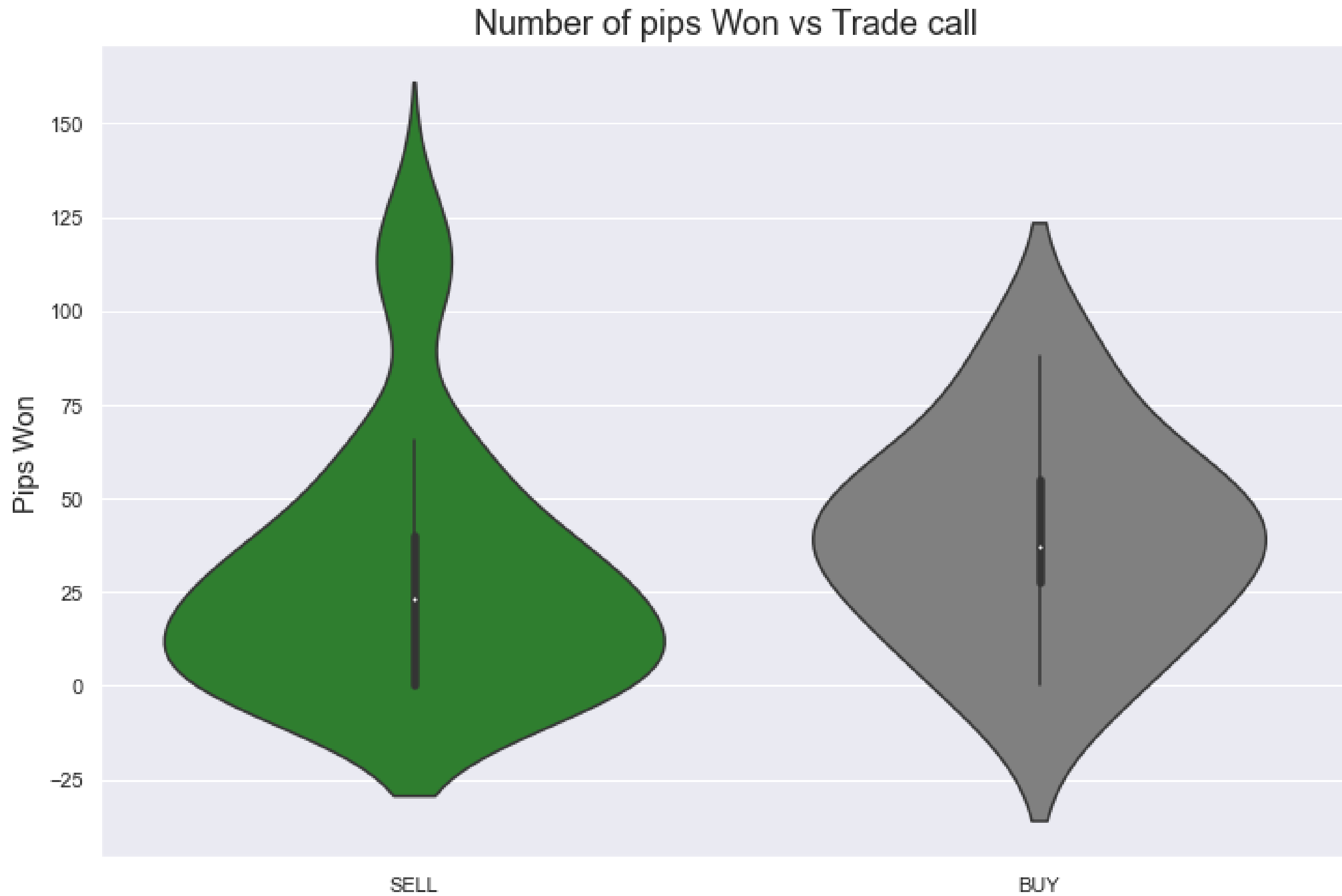
What trade call (Buy or Sell) gave better pip returns?

In [184…
```
df1['Trade'].value_counts()
```

Out[184]:
```
SELL    65
BUY      8
Name: Trade, dtype: int64
```

In [179…
```
#Violin plot comparing home ownership and loan amount
sns.violinplot(data = df1, x='Trade', y= 'no_of_pips_won', palette=sns.color_palette(["f
plt.title('Number of pips Won vs Trade call', fontsize = font_title)
plt.ylabel('Pips Won', fontsize = font_labels)
plt.xlabel(' ', fontsize = font_labels);
```

Out[179]:
```
Text(0.5, 0, ' ')
```



The sell call is multimodal, meaning the data distribution has more than one data cluster, compared to the buy call which is unimodal. The sell call has values with higher pips won compared to the buy call but this may be due to the fact that sell calls were significantly more than buy calls. However, i noticed that the median buy call was higher than the median sell call. This means that relatively, half of pips won from the buy call had higher values than half of pips won from the sell call.

What results were observed within the buy and sell trade calls?

In [185…

```
# Cross tabulation between Trade and result
CrosstabResult=pd.crosstab(index=df['Trade'],columns=df['result'])
print(CrosstabResult)

# Grouped bar chart between Trade results and Trade calls
CrosstabResult.plot.bar(figsize=(7,4), rot=0, color=sns.color_palette(["forestgreen",'gr
plt.title('Trade results vs Trade call', fontsize = font_title)
plt.ylabel('Count', fontsize = font_labels)
plt.xlabel(' ', fontsize = font_labels);
```

```
result  successful  unsuccessful
Trade
BUY              7             1
SELL            46            19
```

Out[185]:

```
Text(0.5, 0, ' ')
```

For both buy and sell trade calls, a higher number of successful trades were observed, compared to the unsuccessful ones.

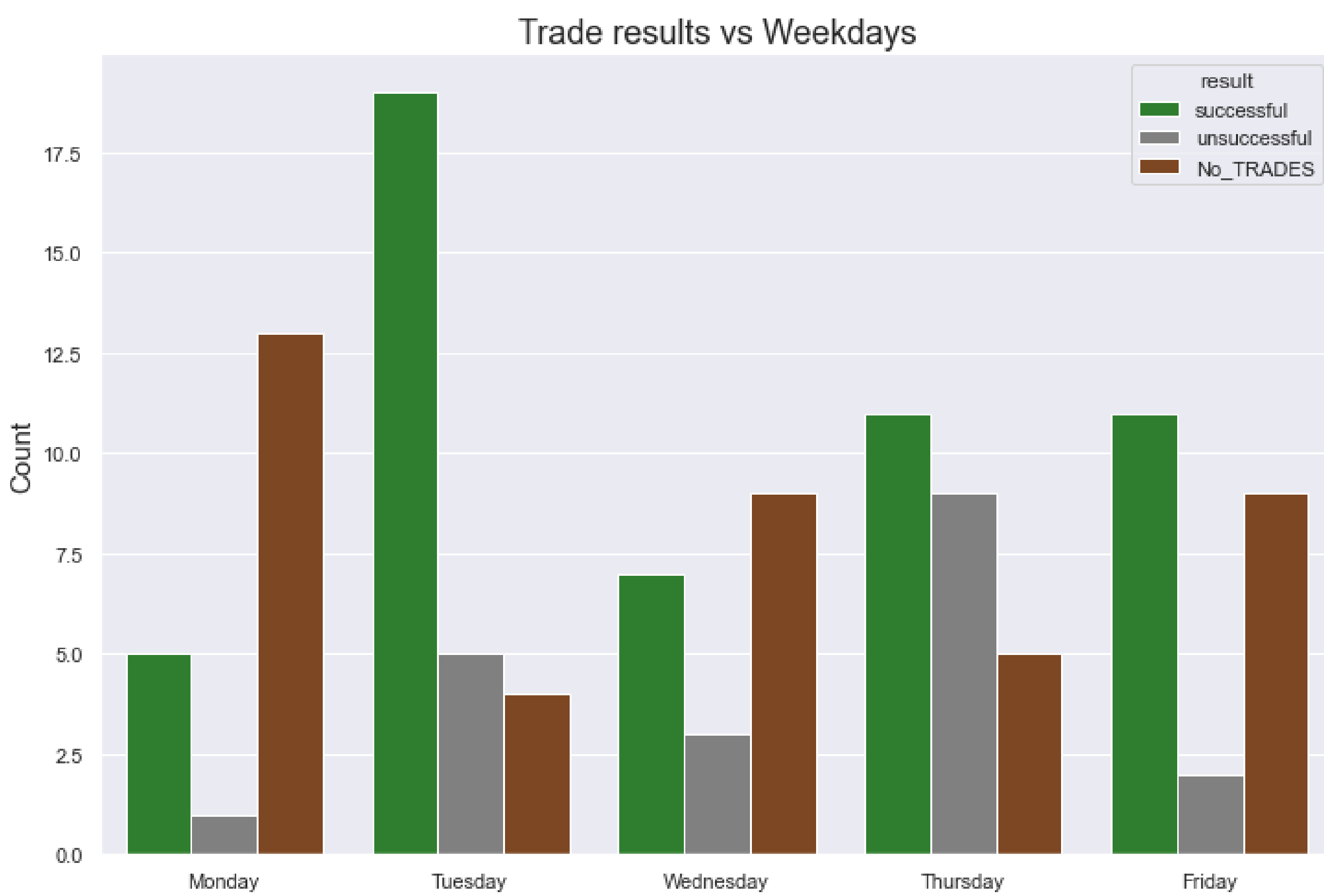
What is the distribution of trade results, across weekdays?

In [186…

```
#Clustered barchart showing the distribution of trade results over weekdays
sns.countplot(data= df_c, x= 'open_day', hue='result', palette=sns.color_palette(["fores
plt.title('Trade results vs Weekdays', fontsize = font_title)
plt.ylabel('Count', fontsize = font_labels)
plt.xlabel(' ', fontsize = font_labels);
```

Out[186]:

```
Text(0.5, 0, ' ')
```

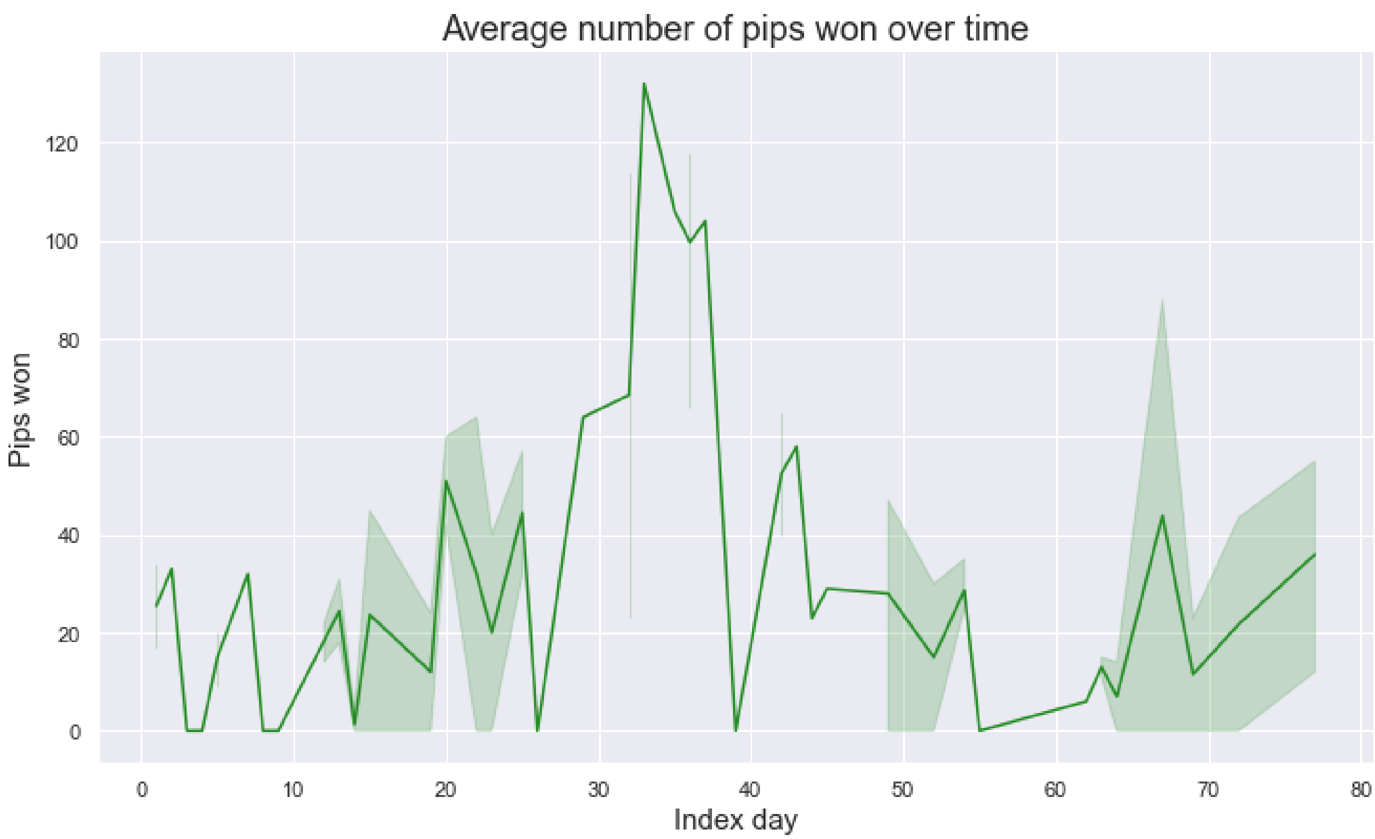


Across all weekdays, except on mondays and wednesdays, successful trades were higher than unsuccessful ones. On mondays and wednesdays, no trades were higher than both the successful and unsuccessful. On mondays, the sum of both the successful and unsuccessful bars did not get to the no trades bar. This confirms that less trades were placed on mondays

What trends can be observed in relation to pips won over the 80 day trade period?

In [187…
```
#Line chart showing average number of pips won over time
sns.relplot(data=df_c, x='Index_day', y='no_of_pips_won', kind="line", height=6, aspect=
plt.title('Average number of pips won over time', fontsize = font_title)
plt.ylabel('Pips won', fontsize = font_labels)
plt.xlabel('Index day', fontsize = font_labels);
```

Out[187]:
```
Text(0.5, 8.959999999999994, 'Index day')
```

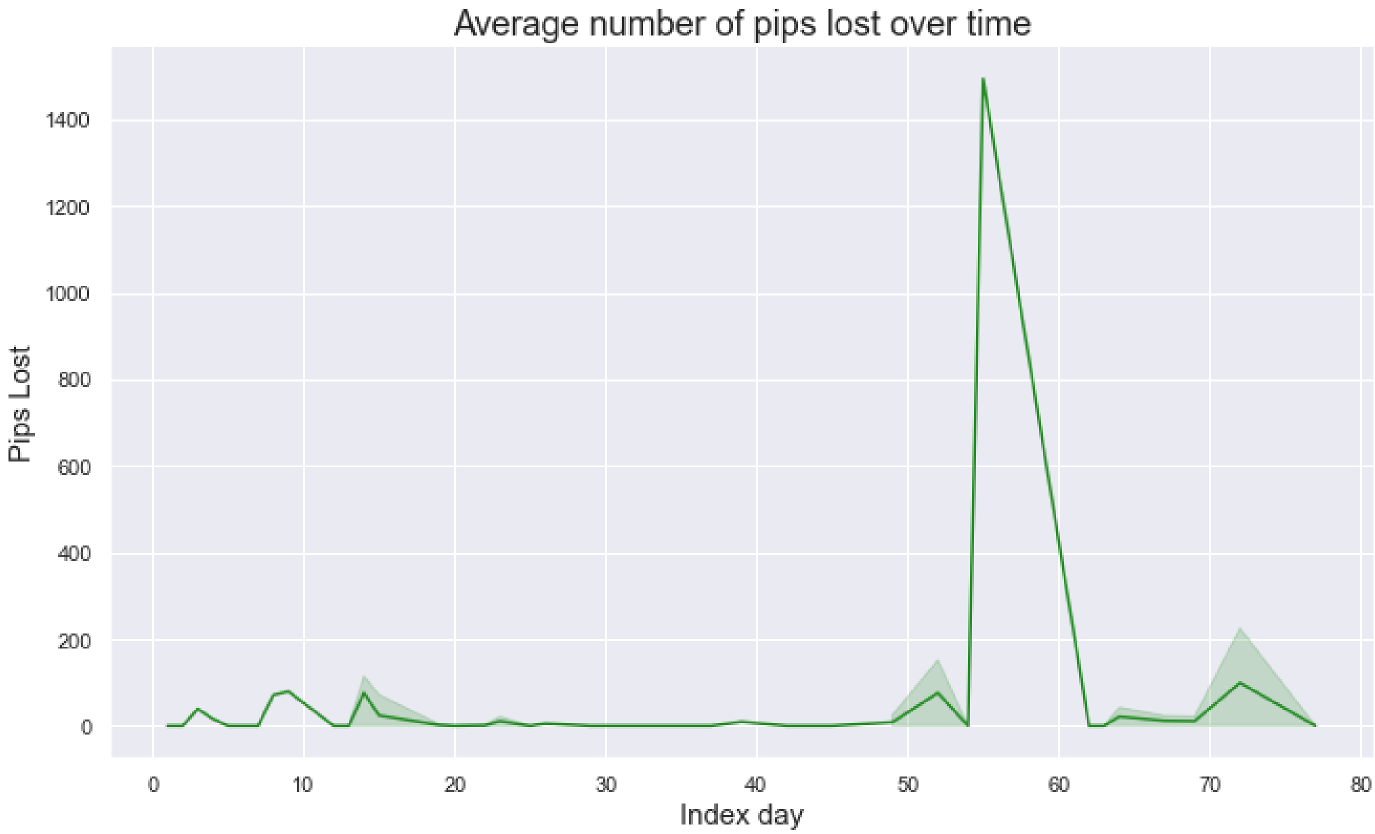


There were several wins but the highest spike was observed between day 30 and day 40. This period corresponds with the last few days of september and continues till mid-october.

What trends can be observed in relation to pips lost over the 80 day trade period?

In [188…
```
#Line chart showing average number of pips lost over time
sns.relplot(data=df_c, x='Index_day', y='no_of_pips_lost', kind="line", height=6, aspect
plt.title('Average number of pips lost over time', fontsize = font_title)
plt.ylabel('Pips Lost', fontsize = font_labels)
plt.xlabel('Index day', fontsize = font_labels)
```

Out[188]:
```
Text(0.5, 8.959999999999994, 'Index day')
```



One significant spike in pip loss was observed, however, the loss was of great magnitude. This was due to the unexpected, significant dollar drop in the market that occurred in the 2022 year end.

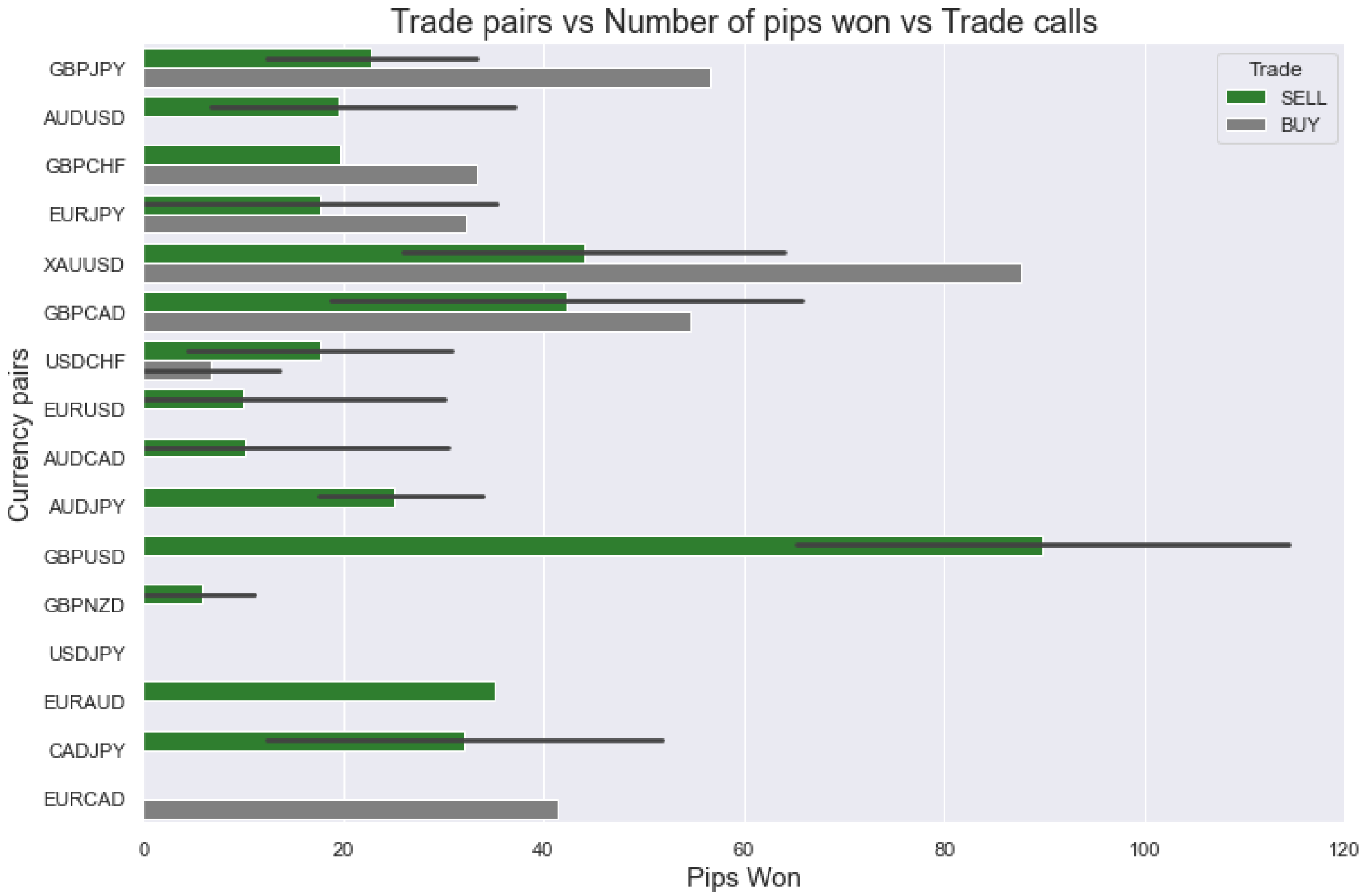
What relationship can be observed between trade pairs, number of pips won and trade calls?

In [189…

```
#Clustered barchart exploring Trade pairs, number of pips won and trade calls
ax= sns.barplot(data = df1, x= 'no_of_pips_won', y= 'Pairs', hue='Trade', palette=sns.co
plt.title('Trade pairs vs Number of pips won vs Trade calls', fontsize = font_title)
plt.ylabel('Currency pairs', fontsize = font_labels)
plt.xlabel('Pips Won', fontsize = font_labels);
```

Out[189]:

```
Text(0.5, 0, 'Pips Won')
```



Interestingly, XAUUSD has more buy pips won than sells. GBPUSD sell pips won are significantly higher than the sell pips of all other currency pairs' EURCAD pips won were only generated from buy trade calls only, while EURUSD, AUDCAD, AUDJPY, GBPUSD, GBPNZD, EURAUD AND CADJPY generated pips from sell trade calls only.

Conclusions

I analysed the prosper loan dataset and discovered the following insights:

- The XAUUSD currency pair was most traded during the 80-days trade period. It accounted for 31.5% of all trades or 23 trades. The GBPJPY pair was next at 13.7% or 10 trades. This is over 50 percent less that the most traded pair.
- Over 57% of trades called were sell trades. About 35% of the time, trades were not called. Buy trades were called only 7% of the whole time.
- Generally, an average trade length of 2 days was observed during the 80 days trade period. Upon further analysis, i observed that although the 'in-a-trade' duration ranged between 0 and 5, two spikes showed two trades which were drawn out for about 25-30days.
- 73% of trades called were successful, while 27% of trades were unsuccessful. This shows an accuracy of about 73% on closed trades.
- 96% of the time, the 0.01 lot size was used. This was because the model account was 200 dollar account.

- This histogram is right skewed, infering that more often than not, pip values of less than 70 were observed, compared to higher values above 70. A spike is also observed at the 110 pip value. This indicates a hiigher count at that level, compared to other larger values.

- A multimodal histogram is observed. Ths shows the maximum trades entered n a day was 5. More frequent was the 4 trade count in a day. The 2 trades a day level also had a significant number of mode values.

- I observed higher chances of opening 3 and 4 trades in a day, compared to closing above 3 trades in a day. A decline is observed when closing from 3 trades and above.

- September recorded the highest number of opened and closed trades. Most opened trades had an average trade length of 0-5 days. This is why september also recorded the highest number of closed trades as well.

- About 32% of all trades were opened on tuesday. This was the highest, compared to other weekdays. This is due to the fact that mondays were usually used to study the trends of the market for the week.

- Most trades were closed on fridays, which marks the end of the trading week. There is no currency pair trading during the weekend. Trading usually resumes on sunday evening.

- No significant relationship was observed between the number of pips won and the number of trades entered

- A negative correlation was observed between the trade length and the number of pips won. This means, the longer the trade length, the lower the number of pips to be won. Longer trade days often led to lower pips, and losses in some cases.

- XAUUSD won the highest number of pips overall, however, the top 75% of all GBPUSD trades gave higher pips than the lower 75% of all XAUUSD trades. The median GBPUSD trade is higher than the median XAUUSD trade, but the top 25% of XAUUSD trades puts it in the lead. USDJPY recorded the least number of pips won overall.

- October recorded the highest number of pips won. 2271.2 pips were won throughout the 80 day trade period

- November recorded the highest number of pips lost. 2702 pips were lost throughout the 80 day trade period.

- The sell call is multimodal, meaning the data distribution has more than one data cluster, compared to the buy call which is unimodal. The sell call has values with higher pips won compared to the buy call but this may be due to the fact that sell calls were significantly more than buy calls. However, i noticed that the median buy call was higher than the median sell call. This means that relatively, half of pips won from the buy call had higher values than half of pips won from the sell call.

- For both buy and sell trade calls, a higher number of successful trades were observed, compared to the unsuccessful ones.

- Across all weekdays, except on mondays and wednesdays, successful trades were higher than unsuccessful ones. On mondays and wednesdays, no trades were higher than both the successful and unsuccessful. On mondays, the sum of both the successful and unsuccessful bars did not get to the no trades bar. This confirms that less trades were placed on mondays

- There were several wins but the highest spike was observed between day 30 and day 40. This period corresponds with the last few days of september and continues till mid-october.
- One significant spike in pip loss was observed, however, the loss was of great magnitude. This was due to the unexpected, significant dollar drop in the market that occurred in the 2022 year end.
- Interestingly, XAUUSD has more buy pips won than sells. GBPUSD sell pips won are significantly higher than the sell pips of all other currency pairs' EURCAD pips won were only generated from buy trade calls only, while EURUSD, AUDCAD, AUDJPY, GBPUSD, GBPNZD, EURAUD AND CADJPY generated pips from sell trade calls only.