Capstone 4 - "Fibre" (EUR/USD) Forex Analysis

Background

The EUR/USD (Euro/US Dollar) currency pair is one of the most widely traded pairs in the foreign exchange (forex) market, representing the exchange rate between the Euro, the official currency of the Eurozone, and the US Dollar, the official currency of the United States. The popularity of this pair stems from the fact that it involves the two largest economies in the world, the Eurozone and the United States, making it a key indicator of global economic health and investor sentiment.

The dataset provided contains financial and trading-related data, including features such as **Trades**, **Gross Profit**, **Gross Loss**, **% Change**, **Profit Factor**, **Winners**, and **Net Profit**. This data represents key performance indicators (KPIs) that are typically used in evaluating the success of trading strategies and financial outcomes over a period of time. The dataset may include both summary statistics for trading strategies and profit metrics, which provide insights into profitability and risks associated with trades.

Given the financial nature of the dataset, it offers an opportunity to apply machine learning methods to derive valuable insights and predictive capabilities. Predictive modeling can aid in identifying trends, minimizing risks, and optimizing decision-making processes in financial trading strategies.

Objective

The primary objective of this project is to apply various machine learning methods to analyze and predict financial performance metrics, such as Net Profit, based on features like Trades, Gross Profit, Gross Loss, % Change, and others. Specifically, the focus will be on the following:

1. **Prediction of Financial Outcomes**: Using supervised learning techniques such as **Regression Analysis** (e.g., Linear Regression, K-Nearest Neighbors, Neural Networks) to predict key financial metrics like Net Profit based on historical data.
2. **Exploration of Relationships Between Variables**: Utilize visualizations such as **pairplots** to explore the relationships between features and their impact on profit outcomes. This will help identify key drivers of profitability in the dataset.
3. **Improvement of Financial Forecasting**: Explore advanced machine learning models, including **Neural Networks**, to improve the accuracy of predictions, uncover hidden patterns, and provide actionable insights for optimizing trading strategies.

The overall goal is to enhance predictive accuracy, which can lead to better decision-making, risk management, and profitability in trading and financial strategy optimization.

Origin of Data

Dataset is downloaded from www.fxblue.com which I have collected the data between 8 Oct 2023 to 4 Aug 2024.

All the "Magic No." is an individual Expert Advisor (EA).

Dataset has the records of 200 rows with 40 columns.

```
# Importing basic libraries
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
import matplotlib.pyplot as plt
import seaborn as sns
```

Loading the Data

```
# Load the dataset
file_path = 'C:/KWM/NTUCLearning Hub/AI ML Developer Capstone Project/Forex.csv'
data = pd.read_csv(file_path)

# Display the first few rows of the dataset
data.head()
```

	Magic	Trades	Gross profit	Gross loss	Net profit	% change	Opening balance	Closing balance	Profit factor	Winners	...	Average winner length (hours)	Average loser length (hours)	Consec winners	Consec losses	Consec profit	Cor l
0	1011000	388	279.79	-342.64	-62.85	-0.64	0	0	0.82	168	...	16.96	17.45	7	9	21.23	-1
1	1011001	229	432.87	-376.76	56.11	0.58	0	0	1.15	111	...	34.37	28.34	7	12	27.25	-3
2	1011002	242	426.83	-433.30	-6.47	-0.07	0	0	0.99	102	...	35.76	25.30	5	11	21.05	-3
3	1011003	321	361.47	-441.40	-79.93	-0.82	0	0	0.82	146	...	23.06	21.62	6	8	19.52	-2
4	1011004	266	310.55	-404.63	-94.08	-0.95	0	0	0.77	118	...	24.35	28.78	9	9	23.14	-4

EDA and Data Preparation

```
1 data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200 entries, 0 to 199
Data columns (total 40 columns):
 #   Column                          Non-Null Count  Dtype
---  ------                          --------------  -----
 0   Magic                           200 non-null    int64
 1   Trades                          200 non-null    int64
 2   Gross profit                    200 non-null    float64
 3   Gross loss                      200 non-null    float64
 4   Net profit                      200 non-null    float64
 5   % change                        200 non-null    float64
 6   Opening balance                 200 non-null    int64
 7   Closing balance                 200 non-null    int64
 8   Profit factor                   200 non-null    float64
 9   Winners                         200 non-null    int64
 10  Winner%                         200 non-null    int64
 11  Losers                          200 non-null    int64
 12  Loser%                          200 non-null    int64
 13  Avg win                         200 non-null    float64
 14  Avg loss                        200 non-null    float64
 15  Avg trade                       200 non-null    float64
 16  Best trade                      200 non-null    float64
 17  Worst trade                     200 non-null    float64
 18  Longest (hours)                 200 non-null    float64
 19  Shortest (hours)                200 non-null    float64
 20  Average length (hours)          200 non-null    float64
 21  Total win pips                  200 non-null    float64
 22  Total loss pips                 200 non-null    float64
 23  Net pips                        200 non-null    float64
 24  Pip profit factor               200 non-null    float64
 25  Avg win pips                    200 non-null    float64
 26  Avg loss pips                   200 non-null    float64
 27  Avg pips per trade              200 non-null    float64
 28  Lots traded                     200 non-null    float64
 29  Avg lots                        200 non-null    float64
 30  Average winner length (hours)   200 non-null    float64
 31  Average loser length (hours)    200 non-null    float64
 32  Consec winners                  200 non-null    int64
 33  Consec losses                   200 non-null    int64
 34  Consec profit                   200 non-null    float64
 35  Consec loss                     200 non-null    float64
 36  Consec win pips                 200 non-null    float64
 37  Consec loss pips                200 non-null    float64
 38  Valley (cash)                   200 non-null    float64
 39  Valley (pips)                   200 non-null    float64
dtypes: float64(30), int64(10)
memory usage: 62.6 KB
```

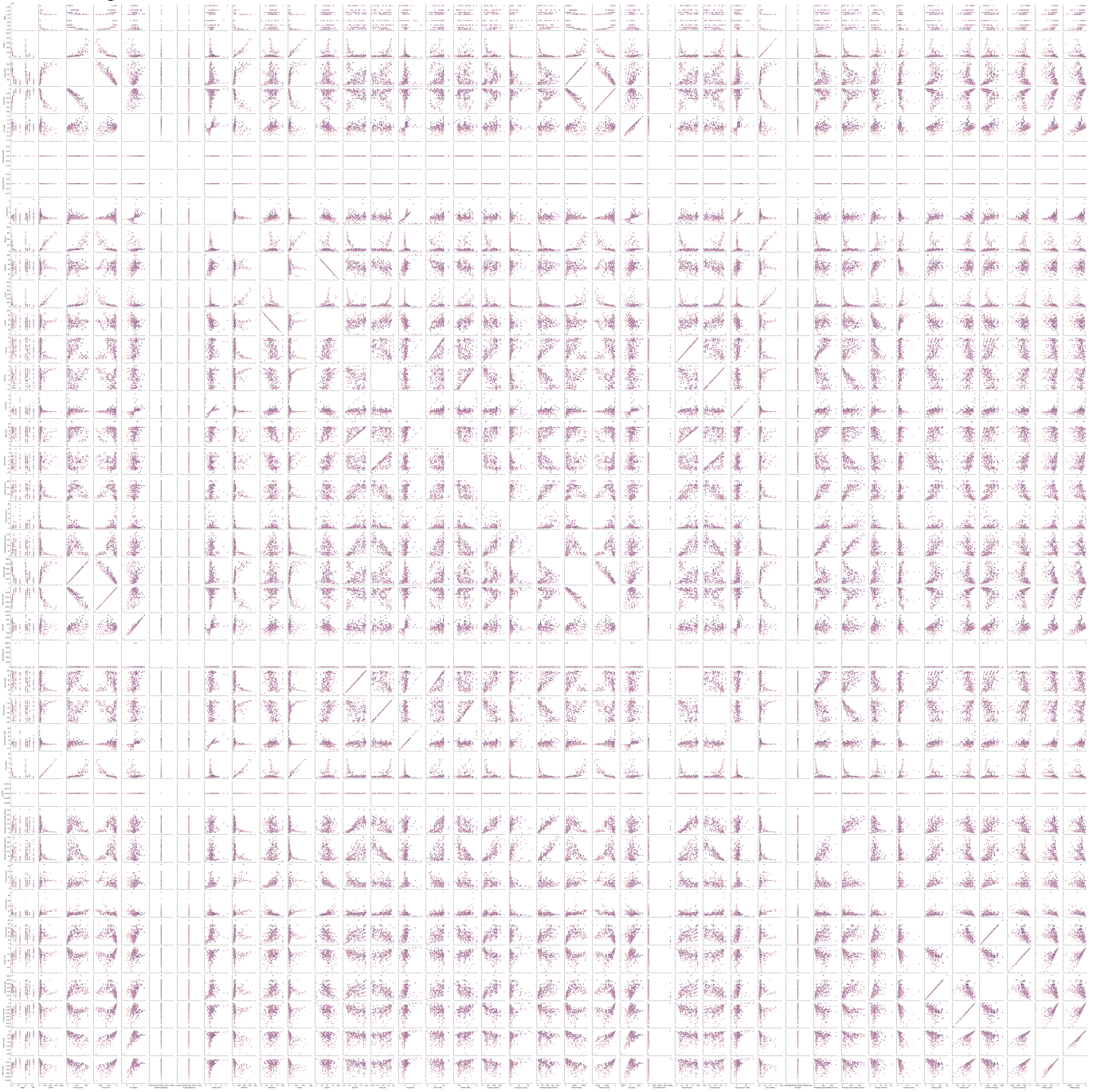
```
1 data.describe()
```

	Magic	Trades	Gross profit	Gross loss	Net profit	% change	Opening balance	Closing balance	Profit factor	Winners	...	Average winner length (hours)
count	2.000000e+02	200.000000	200.000000	200.000000	200.000000	200.000000	200.0	200.0	200.000000	200.000000	...	200.000000
mean	1.723504e+06	105.930000	152.990950	-154.669500	-1.678550	-0.016650	0.0	0.0	1.061200	50.250000	...	76.581150
std	5.451810e+05	161.929338	122.821435	135.082968	44.673946	0.452783	0.0	0.0	0.537034	73.684629	...	54.656456
min	1.011000e+06	1.000000	0.000000	-518.140000	-150.200000	-1.530000	0.0	0.0	0.000000	0.000000	...	0.000000
25%	1.151752e+06	23.000000	44.527500	-220.530000	-23.112500	-0.235000	0.0	0.0	0.790000	10.000000	...	26.385000
50%	1.806504e+06	41.500000	114.550000	-103.520000	0.140000	0.000000	0.0	0.0	0.990000	21.000000	...	72.005000
75%	2.151257e+06	88.000000	235.732500	-48.350000	22.445000	0.225000	0.0	0.0	1.242500	37.250000	...	116.970000
max	2.602009e+06	990.000000	460.400000	0.000000	138.800000	1.420000	0.0	0.0	4.290000	395.000000	...	259.700000

8 rows × 40 columns

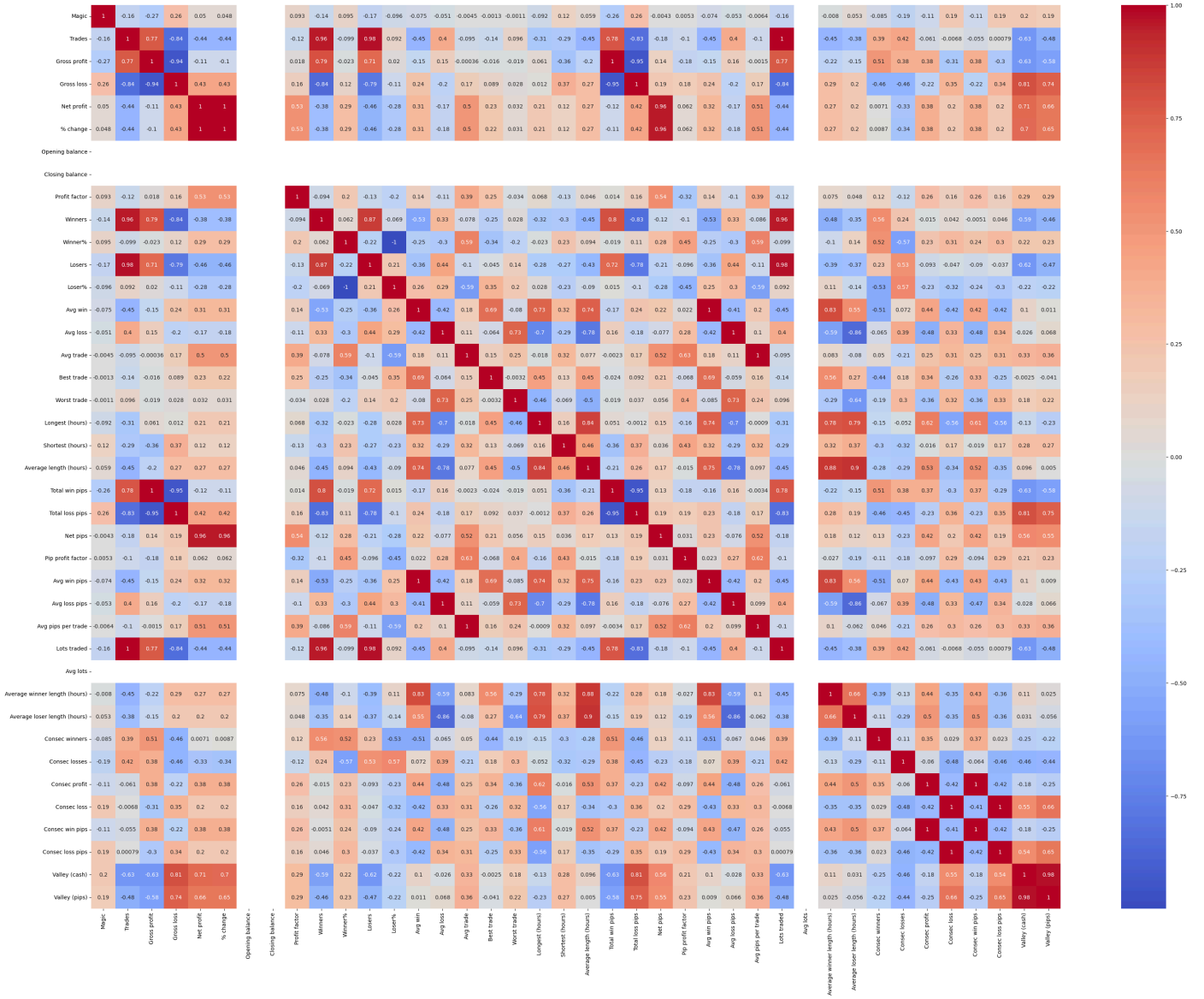
```
sns.pairplot(data,hue='Net profit')
```

<seaborn.axisgrid.PairGrid at 0x141f22d3910>



```
1 corrdf = data.corr()
2 plt.figure(figsize=(40,30))
3 sns.heatmap(corrdf, annot=True, cmap="coolwarm")
```

<Axes: >


Magic
Trades
Gross profit
Gross loss
Net profit
% change
Opening balance
Closing balance
Profit factor
Winners
Winner%
Losers
Loser%
Avg win
Avg loss
Avg trade
Best trade
Worst trade
Longest (hours)
Shortest (hours)
Average length (hours)
Total win pips
Total loss pips
Net pips
Pip profit factor
Avg win pips
Avg loss pips
Avg pips per trade
Lots traded
Avg lots
Average winner length (hours)
Average loser length (hours)
Consec winners
Consec losses
Consec profit
Consec loss
Consec win pips
Consec loss pips
Valley (cash)
Valley (pips)
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75

Remove some of the columns

```
# Define the target variable (e.g., 'Net profit')
target = 'Net profit'

# Select features (excluding the target variable and any non-numeric columns)
features = data.drop(columns=['Average loser length (hours)', 'Average winner length (hours)', 'Avg pips per trade', 'Net pips', 'Magic',

# Convert all features to numeric if needed (useful if there are categorical variables)
features = features.apply(pd.to_numeric, errors='coerce')

# Drop any rows with missing values (optional, based on your data)
features = features.dropna()
```

```
features.head()
```

	Trades	Net profit	Profit factor	Winner%	Loser%	Avg trade	Best trade	Worst trade	Average length (hours)	Total win pips	Total loss pips	Pip profit factor	Avg win pips	Avg loss pips	Consec win pips	Consec loss pips
0	388	-62.85	0.82	43	56	-0.16	7.21	-7.59	17.10	2929.2	-3260.1	0.898500	17.4	-15.0	218.4	-147.9
1	229	56.11	1.15	48	52	0.25	4.10	-3.72	31.26	4434.9	-3675.3	1.206677	40.0	-31.1	279.6	-383.2
2	242	-6.47	0.99	42	58	-0.03	4.41	-3.80	29.71	4373.5	-4216.8	1.037161	42.9	-30.1	214.7	-330.0
3	321	-79.93	0.82	45	54	-0.25	4.89	-5.07	22.21	3745.4	-4273.8	0.876363	25.7	-24.6	198.7	-279.6

```
features.describe()
```

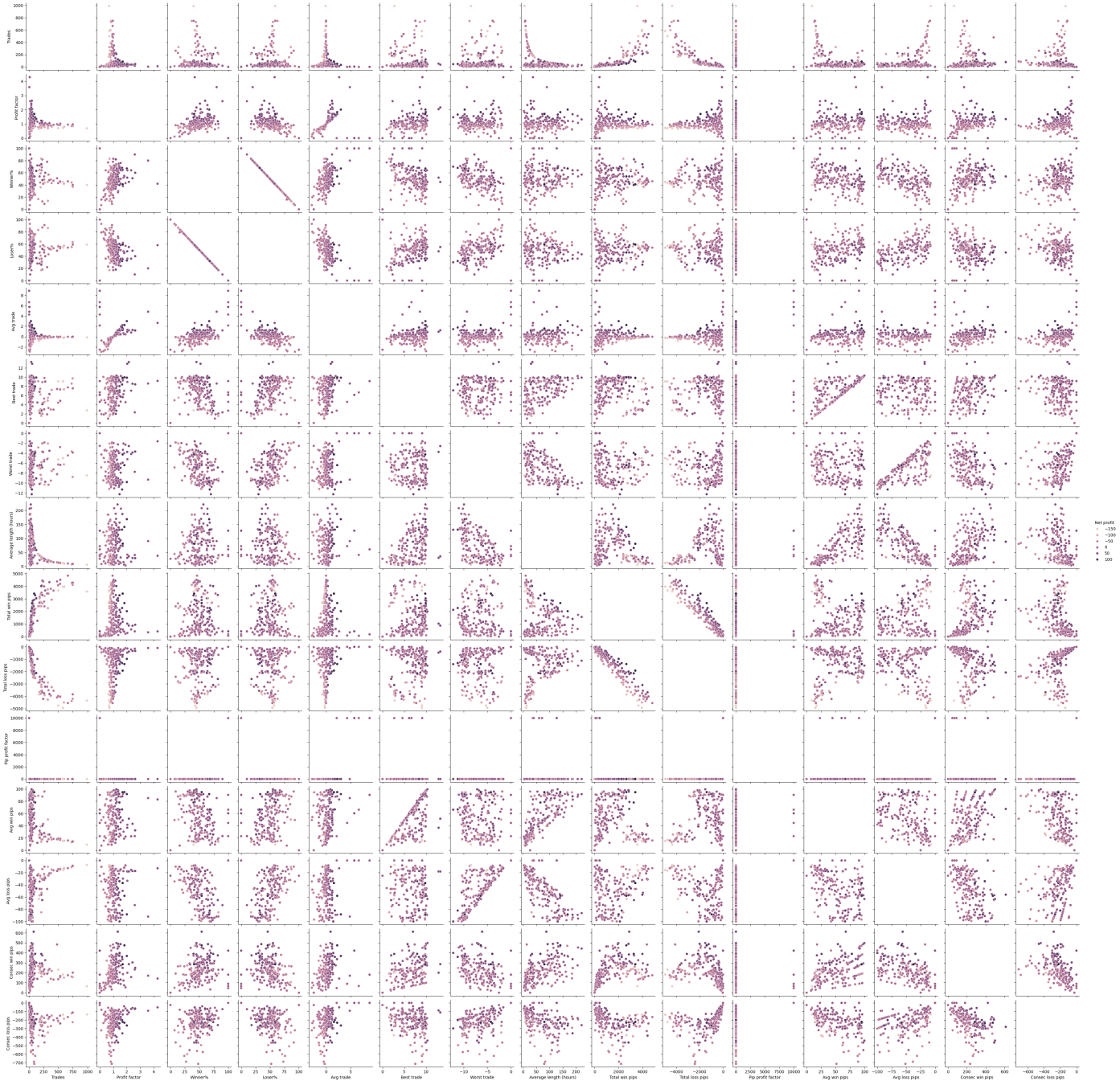
	Trades	Net profit	Profit factor	Winner%	Loser%	Avg trade	Best trade	Worst trade	Average length (hours)	Total win pips	Total loss pips
count	200.000000	200.000000	200.000000	200.000000	200.000000	200.00000	200.000000	200.000000	200.000000	200.000000	200.000000
mean	105.930000	-1.678550	1.061200	50.060000	49.865000	0.18385	6.934450	-6.815900	72.539800	1581.964000	-1499.750000
std	161.929338	44.673946	0.537034	17.701784	17.640086	1.37726	2.593167	2.748002	52.397489	1279.881718	1292.797626
min	1.000000	-150.200000	0.000000	0.000000	0.000000	-2.94000	0.000000	-12.250000	3.300000	0.000000	-4933.600000
25%	23.000000	-23.112500	0.790000	41.000000	37.750000	-0.40500	4.920000	-9.395000	26.960000	458.750000	-2156.225000
50%	41.500000	0.140000	0.990000	48.000000	52.000000	0.00000	7.405000	-6.785000	62.100000	1178.900000	-1027.450000
75%	88.000000	22.445000	1.242500	62.250000	59.000000	0.67000	9.115000	-4.700000	112.477500	2410.150000	-462.075000

```
features.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200 entries, 0 to 199
Data columns (total 16 columns):
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   Trades                 200 non-null    int64
 1   Net profit             200 non-null    float64
 2   Profit factor          200 non-null    float64
 3   Winner%                200 non-null    int64
 4   Loser%                 200 non-null    int64
 5   Avg trade              200 non-null    float64
 6   Best trade             200 non-null    float64
 7   Worst trade            200 non-null    float64
 8   Average length (hours) 200 non-null    float64
 9   Total win pips         200 non-null    float64
 10  Total loss pips        200 non-null    float64
 11  Pip profit factor      200 non-null    float64
 12  Avg win pips           200 non-null    float64
 13  Avg loss pips          200 non-null    float64
 14  Consec win pips        200 non-null    float64
 15  Consec loss pips       200 non-null    float64
dtypes: float64(13), int64(3)
memory usage: 25.1 KB
```

```
1 sns.pairplot(features,hue='Net profit')
```

<seaborn.axisgrid.PairGrid at 0x141bb24fd10>



```
1 corrdf = features.corr()
2 plt.figure(figsize=(30,20))
3 sns.heatmap(corrdf, annot=True, cmap="coolwarm")
```

```
<Axes: >
```


```
# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(features, data[target], test_size=0.2, random_state=42)
```


Linear Regression Model


```
# Initialize the Linear Regression model
model = LinearRegression()

# Train the model on the training data
model.fit(X_train, y_train)

# Get the model's coefficients
print("Coefficients:", model.coef_)
print("Intercept:", model.intercept_)
```

```
Coefficients: [-1.94127432e-15  1.00000000e+00  2.57164994e-14 -4.97990519e-14
 -5.51045072e-14  1.81667403e-14 -7.15899562e-15 -1.45001997e-14
```