Machine Learning and Data Science Blueprints for Finance

Data science and Artificial Intelligence Society

Seonggeun Cho

Project Proposal

Topic: Machine Learning

Description: Examine ML concepts and case studies in supervised, unsupervised, and reinforcement learning, along with natural language processing (NLP), such as trading strategies, derivative pricing, portfolio management, etc

Expected Duration: 10 weeks

Bitcoin Signal Detection

Predict whether the current signal is buy or sell depending on the short term vs. long term price.

Main shortcomings of Bitcoin transaction = Variability.

- Cryptocurrency market opens 24/7
- Auto trading algorithm or trading bot might be helpful

Timeline

Exploratory Data Analysis

Descriptive Statistics

Data Preparation

Data Cleaning and Labeling

Feature Engineering

Calculate Technical Indicators

Model Comparison and Selection

Model Tuning and Grid Search

Model Evaluation and Analysis

1. Exploratory Data Analysis

	Timestamp	Open	High	Low	Close	Volume_(BTC)	Volume_(Currency)	Weighted_Price
count	2.841e+06	1.651e+06	1.651e+06	1.651e+06	1.651e+06	1.651e+06	1.651e+06	1.651e+06
mean	1.411e+09	4.959e+02	4.962e+02	4.955e+02	4.959e+02	1.188e+01	5.316e+03	4.959e+02
std	4.938e+07	3.642e+02	3.645e+02	3.639e+02	3.643e+02	4.094e+01	1.998e+04	3.642e+02
min	1.325e+09	3.800e+00	3.800e+00	1.500e+00	1.500e+00	0.000e+00	0.000e+00	3.800e+00
25%	1.368e+09	2.399e+02	2.400e+02	2.398e+02	2.399e+02	3.828e-01	1.240e+02	2.399e+02
50%	1.411e+09	4.200e+02	4.200e+02	4.199e+02	4.200e+02	1.823e+00	6.146e+02	4.200e+02
75%	1.454e+09	6.410e+02	6.417e+02	6.402e+02	6.410e+02	8.028e+00	3.108e+03	6.410e+02
max	1.496e+09	2.755e+03	2.760e+03	2.752e+03	2.755e+03	5.854e+03	1.866e+06	2.754e+03

2. Data Preparation

Clean the data by filling the *NaNs* with the last available values.

Attach a label to each movement:

- **1** if the signal is that short term price will go up as compared to the long term.
- **0** if the signal is that short term price will go down as compared to the long term.

```
# Define short moving average as the average of the past 10-day close price
dataset['short_mavg'] = dataset['Close'].rolling(window=10, min_periods=1, center=False).mean()

# Define long moving average as the average of the past 60-day close price
dataset['long_mavg'] = dataset['Close'].rolling(window=60, min_periods=1, center=False).mean()

# Create a signal by comparing short_mavg and long_mavg
dataset['signal'] = np.where(dataset['short_mavg'] > dataset['long_mavg'], 1.0, 0.0)
```

3. Feature Engineering

Calculate the following technical indicators:

Moving Average

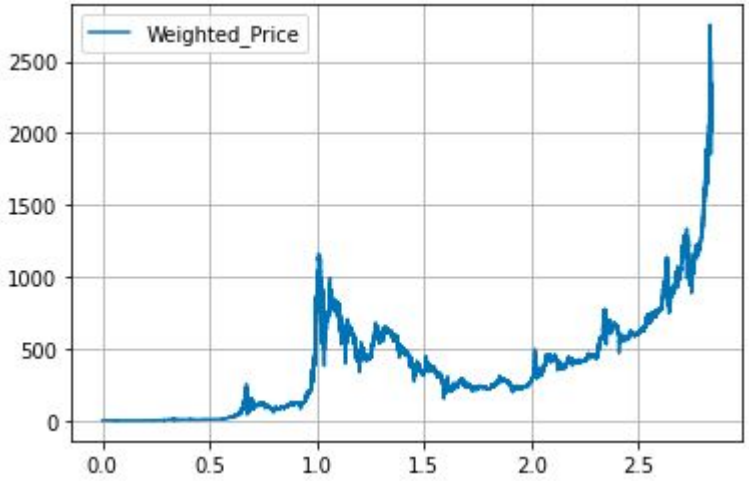
Stochastic Oscillator %K and %D

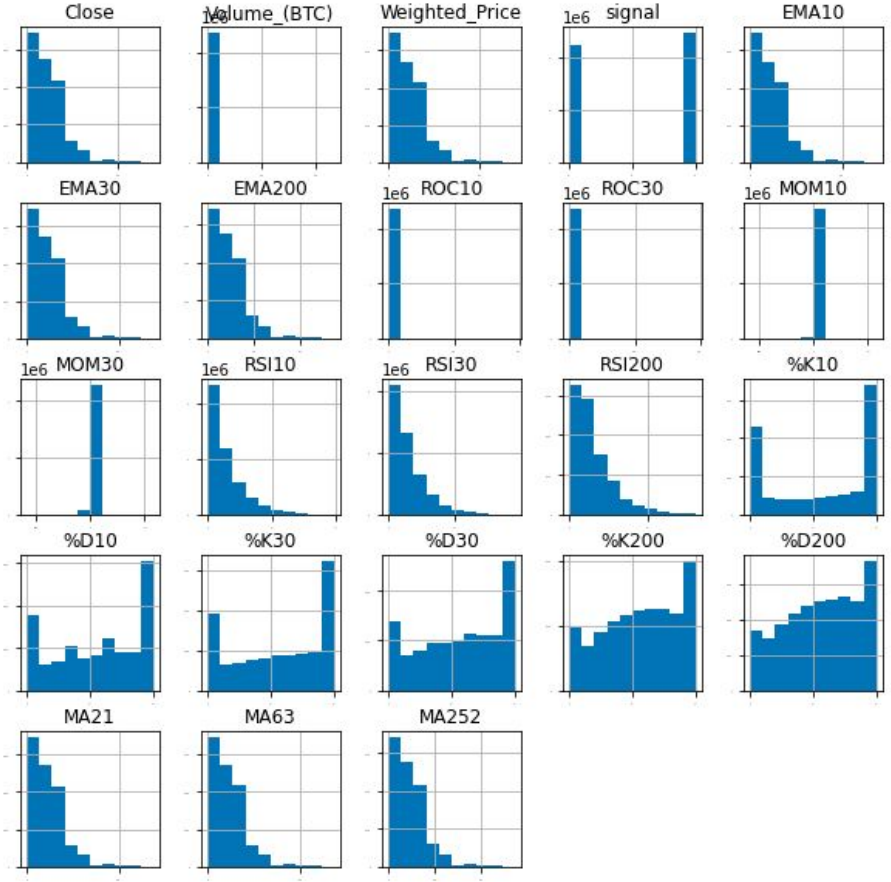
Relative Strength Index(RSI)

Rate Of Change(ROC)

Momentum (MOM)

Visualization





Recap

1. **Exploratory Data Analysis** - Looked over dataset via descriptive statistics.

2. **Data Preparation** - Filled out NaNs and attached a label

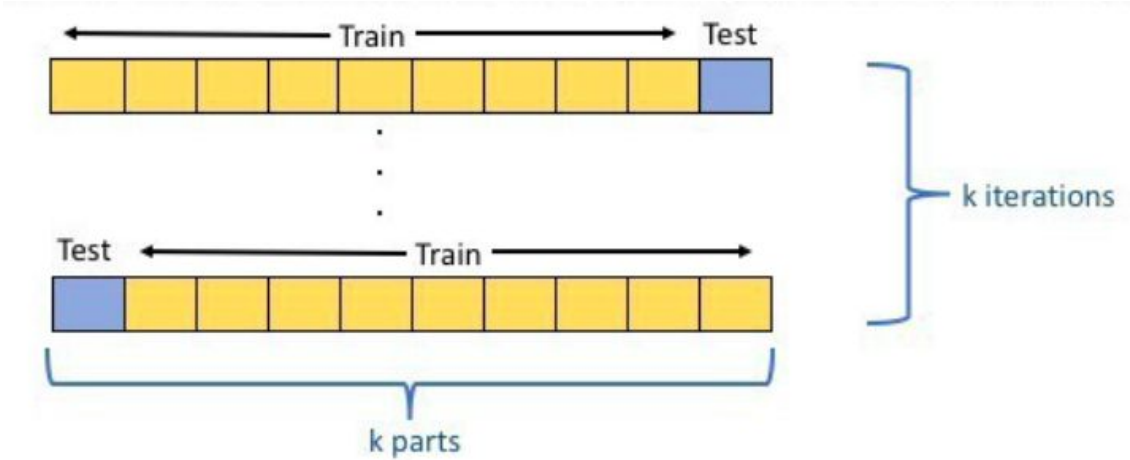
3. **Feature Engineering** - Calculated the technical indicators (EMA, %K, ...)

4. Model Comparison and Selection

K-Folds cross validation.

10 Folds.

Metric = accuracy



Splitting the Dataset

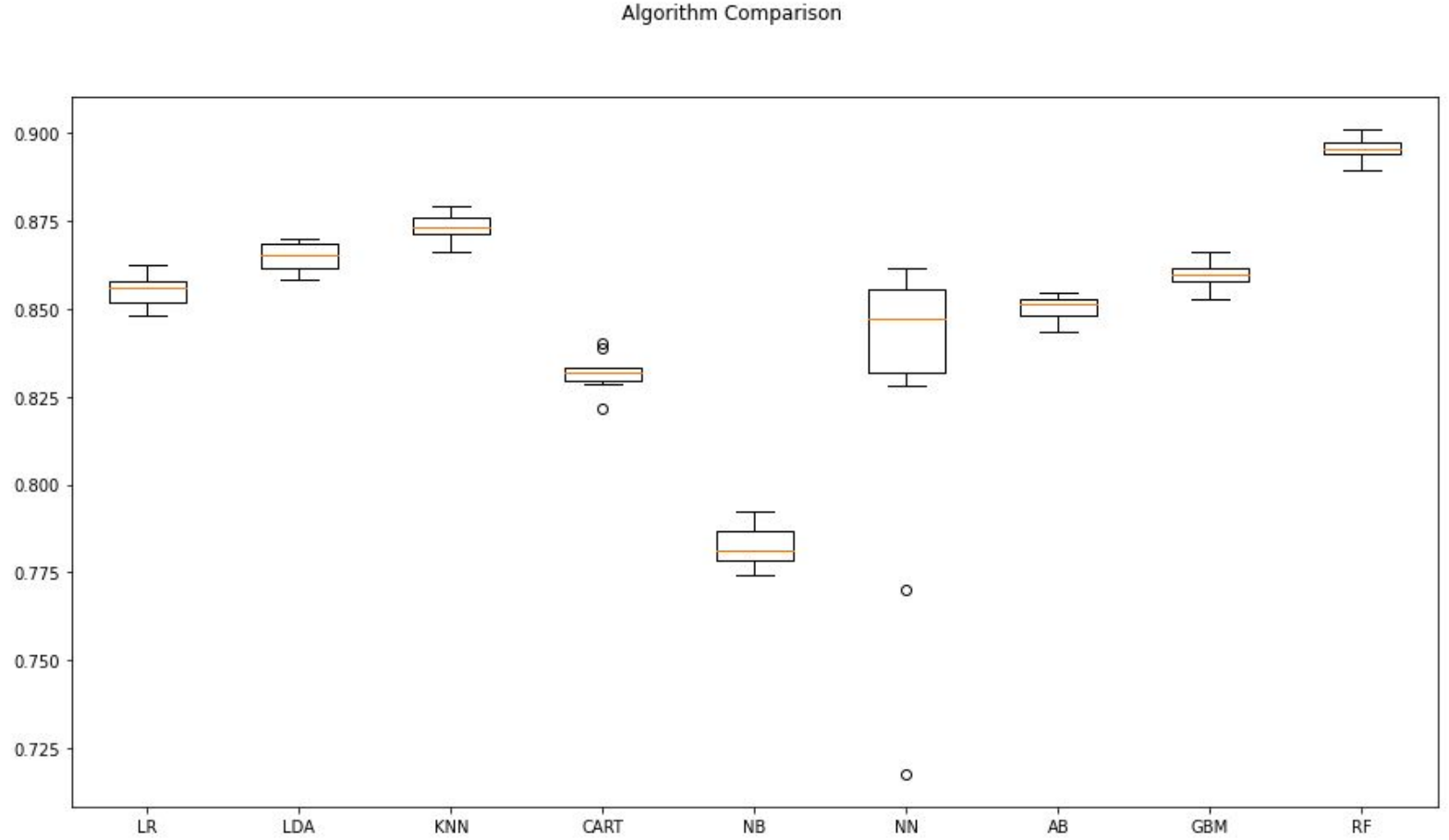
```
# split out validation dataset
subset_dataset = dataset.iloc[-100000:]
Y = subset_dataset["signal"]
X = subset_dataset.loc[:, dataset.columns != 'signal']
validation_size = 0.2
seed = 1
X_train, X_validation, Y_train, Y_validation = train_test_split(X, Y, test_size=validation_size, random_state=1)
```

KFold Cross Validation

Comparison on 9 types of model

```
results = []
names = []
for name, model in models:
  kfold = KFold(n_splits=num_folds, random_state=seed, shuffle=True)
  cv_results = cross_val_score(model, X_train, Y_train, cv=kfold)
  results.append(cv_results)
  names.append(name)
  msg = "%s: %f (%f)" % (name, cv_results.mean(), cv_results.std())
  print(msg)
```

```
LR: 0.855287 (0.004591)
LDA: 0.864975 (0.003834)
KNN: 0.873562 (0.003593)
CART: 0.831763 (0.004924)
NB: 0.782425 (0.005360)
NN: 0.828375 (0.044885)
AB: 0.850400 (0.003634)
GBM: 0.859513 (0.003644)
RF: 0.895288 (0.003421)
```


Algorithm Comparison
0.900
0.875
0.850
0.825
0.800
0.775
0.750
0.725
LR
LDA
KNN
CART
NB
NN
AB
GBM
RF

5. Model Tuning and Grid Search

Grid search: Compare every number of cases and find combinations of parameter with the best performance

Perform grid search on parameter(criterion, n_estimators, max_depth)

Random Forest Classifier

Ensemble - bagging

- Earn variability by limiting dataset and the features while training the Tree

Offset the shortcomings of Decision Tree Classifier

Conclusion is obtained by collecting the results from multiple Trees constructed through the training

Grid Search

```
scaler = StandardScaler().fit(X_train)
rescaledX = scaler.transform(X_train)
n_estimators = [20,80]
max_depth= [5,10]
criterion = ["gini","entropy"]
param_grid = dict(n_estimators=n_estimators, max_depth=max_depth, criterion = criterion )
model = RandomForestClassifier(n_jobs=-1)
kfold = KFold(n_splits=num_folds, random_state=seed, shuffle=True)
grid = GridSearchCV(estimator=model, param_grid=param_grid, scoring=scoring, cv=kfold)
grid_result = grid.fit(rescaledX, Y_train)
```

6. Model Evaluation and Analysis

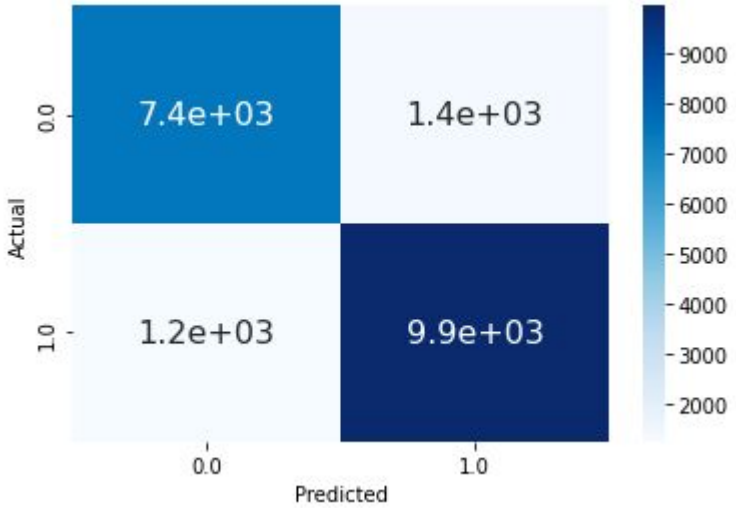
Final accuracy: **86.855%**

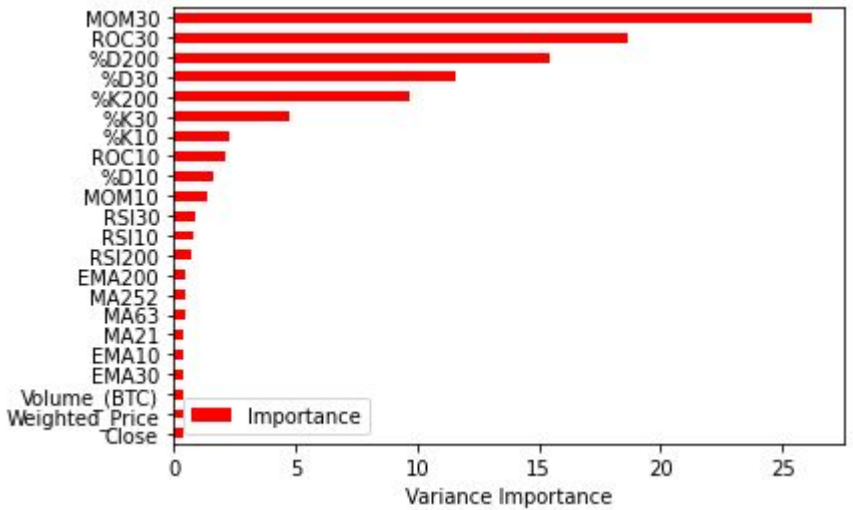
Recall = 84%, 89%

F1-score = 85%, 88%

	precision	recall	f1-score	support
0.0	0.86	0.84	0.85	8818
1.0	0.88	0.89	0.88	11182
accuracy			0.87	20000
macro avg	0.87	0.87	0.87	20000
weighted avg	0.87	0.87	0.87	20000

Confusion Matrix & Feature Importance





Thank you.