Golden_Death_Cross

September 29, 2021

1 Golden Cross and Death Cross

https://www.investopedia.com/terms/g/goldencross.asp

https://www.investopedia.com/terms/d/deathcross.asp

[1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

import warnings
warnings.filterwarnings("ignore")

import yfinance as yf
yf.pdr_override()
```

[2]:

```
start = '2014-01-01'
end = '2019-01-01'

symbol = 'AAPL'

df = yf.download(symbol, start=start, end=end)
```

```
[*********************100%***********************]  1 of 1 completed
```

[3]:

```
df.head()
```

[3]:

```
             Adj Close      Close       High        Low       Open     Volume
Date
2014-01-02  71.107201  79.018570  79.575714  78.860001  79.382858   58671200
2014-01-03  69.545288  77.282860  79.099998  77.204285  78.980003   98116900
2014-01-06  69.924515  77.704285  78.114288  76.228569  76.778572  103152700
2014-01-07  69.424438  77.148575  77.994286  76.845711  77.760002   79302300
2014-01-08  69.864105  77.637146  77.937141  76.955711  76.972855   64632400
```

[4]:

```
df['% Change'] = df['Adj Close'].pct_change() # pct_change : percent profit rate
df.head()
```

```
[4]:             Adj Close      Close       High        Low       Open     Volume  \
     Date
     2014-01-02  71.107201  79.018570  79.575714  78.860001  79.382858   58671200
     2014-01-03  69.545288  77.282860  79.099998  77.204285  78.980003   98116900
     2014-01-06  69.924515  77.704285  78.114288  76.228569  76.778572  103152700
     2014-01-07  69.424438  77.148575  77.994286  76.845711  77.760002   79302300
     2014-01-08  69.864105  77.637146  77.937141  76.955711  76.972855   64632400

                 % Change
     Date
     2014-01-02       NaN
     2014-01-03 -0.021966
     2014-01-06  0.005453
     2014-01-07 -0.007152
     2014-01-08  0.006333
```

```
[5]: df['P_AdjClose'] = df['Adj Close'].shift(1) # P_Adj Close : previous Close
```

```
[6]: df['L_Profit'] = np.log(df['Adj Close'] / df['P_AdjClose']) # L_Profit : Log␣
      ↪Profit Rate
```

```
[7]: df['MA_50'] = df['Adj Close'].rolling(center=False, window=50).mean()
     df['MA_200'] = df['Adj Close'].rolling(center=False, window=200).mean()
     df['diff'] = df['MA_50'] - df['MA_200']

     df = df[['Volume', 'Adj Close', 'MA_50', 'MA_200', 'diff']]
     df.head(20)
```

```
[7]:                Volume  Adj Close  MA_50  MA_200  diff
     Date
     2014-01-02   58671200  71.107201    NaN     NaN   NaN
     2014-01-03   98116900  69.545288    NaN     NaN   NaN
     2014-01-06  103152700  69.924515    NaN     NaN   NaN
     2014-01-07   79302300  69.424438    NaN     NaN   NaN
     2014-01-08   64632400  69.864105    NaN     NaN   NaN
     2014-01-09   69787200  68.971939    NaN     NaN   NaN
     2014-01-10   76244000  68.511696    NaN     NaN   NaN
     2014-01-13   94623200  68.870354    NaN     NaN   NaN
     2014-01-14   83140400  70.240776    NaN     NaN   NaN
     2014-01-15   97909700  71.651016    NaN     NaN   NaN
     2014-01-16   57319500  71.251190    NaN     NaN   NaN
     2014-01-17  106684900  69.505417    NaN     NaN   NaN
     2014-01-21   82131700  70.585274    NaN     NaN   NaN
     2014-01-22   94996300  70.898949    NaN     NaN   NaN
     2014-01-23  100809800  71.499313    NaN     NaN   NaN
     2014-01-24  107338700  70.199638    NaN     NaN   NaN
     2014-01-27  138719700  70.769104    NaN     NaN   NaN
```

```
2014-01-28  266380800  65.112747    NaN      NaN    NaN
2014-01-29  125702500  64.373535    NaN      NaN    NaN
2014-01-30  169625400  64.248833    NaN      NaN    NaN
```

[8]:

```
prev_key = prev_val = 0

for key, val in df['diff'].iteritems():
  if val == 0:
    continue
  if val * prev_val < 0 and val > prev_val:
    print('[Golden]', key, val, df['Adj Close'][key])
  if val * prev_val < 0 and val < prev_val:
    print('[Death]', key, val, df['Adj Close'][key])
  prev_key, prev_val = key, val
```

```
[Death] 2015-08-28 00:00:00 -0.21629188537598054 105.41515350341797
[Golden] 2016-08-30 00:00:00 0.02425605773925099 100.74073791503906
[Death] 2018-12-21 00:00:00 -0.44662147521972884 148.49879455566406
```

1.0.1 50-day Moving Average goes 'under' 200-day Moving Average is a "Death Cross."

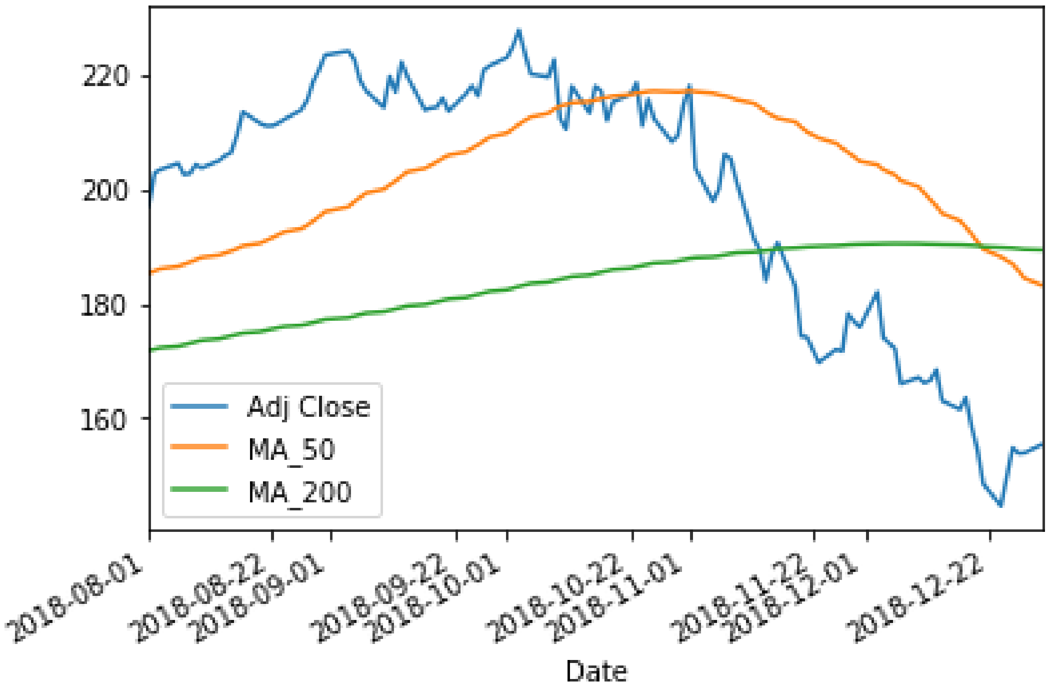
1.0.2 50-day Moving Average goes 'over' 200-day Moving Average is a "Golden Cross."

[9]:

```
df[['Adj Close', 'MA_50', 'MA_200']]['2018-08':].plot()
```

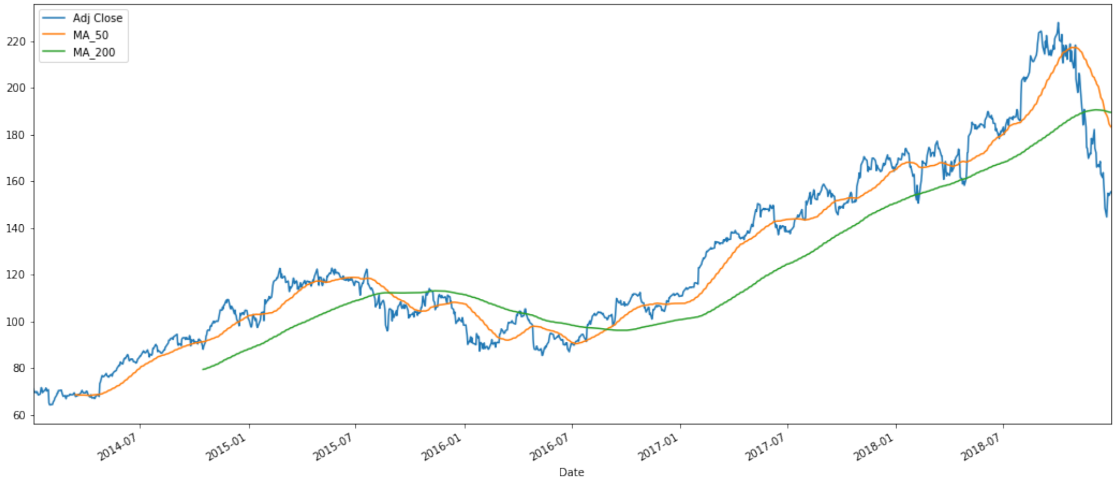
[9]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1525cc72278>
```



[10]:

```
ax = df[['Adj Close', 'MA_50', 'MA_200']].plot(figsize=(18, 8))

prev_key = prev_val = 0

for key, val in df['diff'].iteritems():
  if val == 0:
    continue

    if val * prev_val < 0 and val > prev_val:
        ax.annotate('Golden', xy=(key, df['MA_200'][key]), xytext=(10,-30),
                    textcoords='offset points',␣
 ↪arrowprops=dict(arrowstyle='-|>'))
    elif val * prev_val < 0 and val < prev_val:
        ax.annotate('Death', xy=(key, df['MA_200'][key]), xytext=(10,30),
                    textcoords='offset points',␣
 ↪arrowprops=dict(arrowstyle='-|>'))

    prev_key, prev_val = key, val
```



1.1 Plot Multi-colors of one line

[11]:

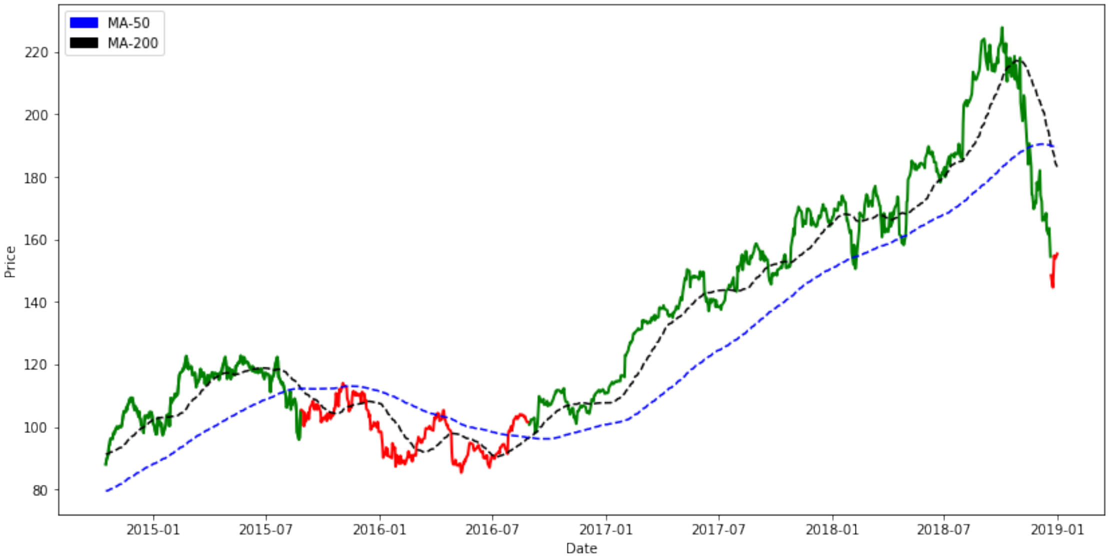
```
df['Label'] = np.where(df['MA_50'] > df['MA_200'], 1, -1)
```

[12]:

```
import matplotlib.patches as mpatches

df = df.dropna(axis=0, how='any')
```

```
fig, ax = plt.subplots(figsize=(14,7))

def plot_colors(group):
    global ax
    color = 'r' if (group['Label'] < 0).all() else 'g'
    lw = 2.0
    ax.plot(group.index, group['Adj Close'], c=color, linewidth=lw)

df.groupby((df['Label'].shift() * df['Label'] < 0).cumsum()).apply(plot_colors)

ax.plot(df.index, df['MA_50'], 'k--', label='MA-50')
ax.plot(df.index, df['MA_200'], 'b--', label='MA-200')
ax.set_ylabel('Price')
ax.set_xlabel('Date')
ma50 = mpatches.Patch(color='blue', label='MA-50')
ma200 = mpatches.Patch(color='black', label='MA-200')
ax.legend(handles=[ma50,ma200])
```

[12]:
```
<matplotlib.legend.Legend at 0x1525edd94e0>
```



1.2 Plot Intersection Points

[13]:
```
short_term = 50
long_term = 200

signals = pd.DataFrame(index=df.index)
signals['position'] = 0.0
```

```
signals['Short_MA'] = df['Adj Close'].rolling(window=short_term, center=False).
 ↪mean()
signals['Long_MA'] = df['Adj Close'].rolling(window=long_term, center=False).
 ↪mean()

signals['position'][short_term:] = np.where(signals['Short_MA'][short_term:]
                                            > signals['Long_MA'][short_term:],␣
 ↪1.0, 0.0)

signals['intersection'] = signals['position'].diff()
```

[14]:

```
print(signals)
```

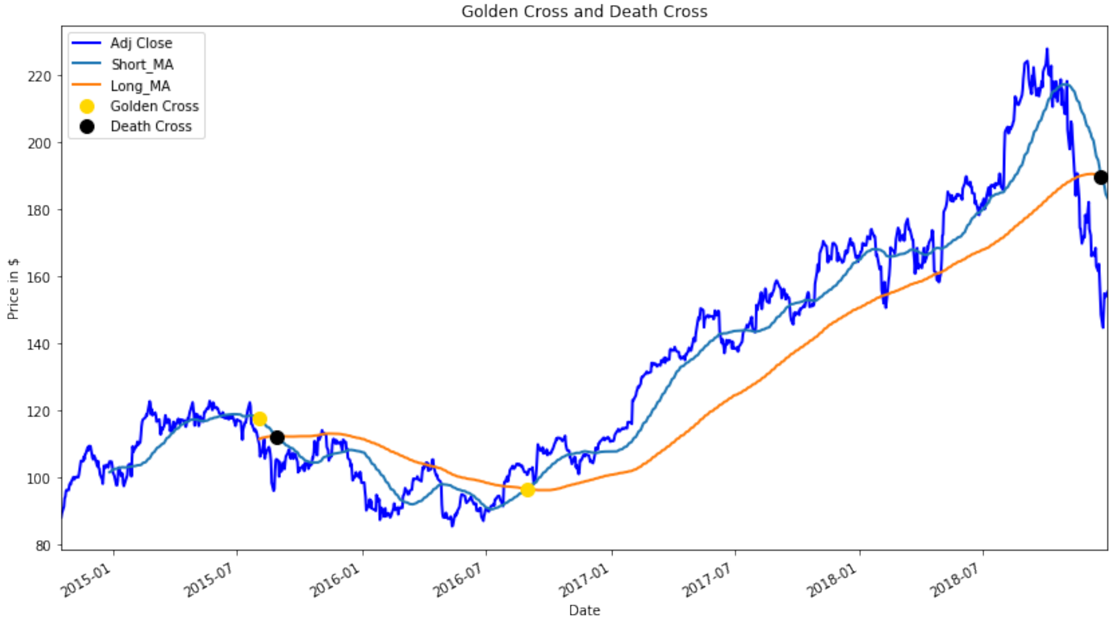
```
            position    Short_MA     Long_MA  intersection
Date
2014-10-16       0.0         NaN         NaN           NaN
2014-10-17       0.0         NaN         NaN           0.0
2014-10-20       0.0         NaN         NaN           0.0
2014-10-21       0.0         NaN         NaN           0.0
2014-10-22       0.0         NaN         NaN           0.0
2014-10-23       0.0         NaN         NaN           0.0
2014-10-24       0.0         NaN         NaN           0.0
2014-10-27       0.0         NaN         NaN           0.0
2014-10-28       0.0         NaN         NaN           0.0
2014-10-29       0.0         NaN         NaN           0.0
2014-10-30       0.0         NaN         NaN           0.0
2014-10-31       0.0         NaN         NaN           0.0
2014-11-03       0.0         NaN         NaN           0.0
2014-11-04       0.0         NaN         NaN           0.0
2014-11-05       0.0         NaN         NaN           0.0
2014-11-06       0.0         NaN         NaN           0.0
2014-11-07       0.0         NaN         NaN           0.0
2014-11-10       0.0         NaN         NaN           0.0
2014-11-11       0.0         NaN         NaN           0.0
2014-11-12       0.0         NaN         NaN           0.0
2014-11-13       0.0         NaN         NaN           0.0
2014-11-14       0.0         NaN         NaN           0.0
2014-11-17       0.0         NaN         NaN           0.0
2014-11-18       0.0         NaN         NaN           0.0
2014-11-19       0.0         NaN         NaN           0.0
2014-11-20       0.0         NaN         NaN           0.0
2014-11-21       0.0         NaN         NaN           0.0
2014-11-24       0.0         NaN         NaN           0.0
2014-11-25       0.0         NaN         NaN           0.0
2014-11-26       0.0         NaN         NaN           0.0
…                …       …        …         …
2018-11-15       1.0  212.989678  189.429142           0.0
```

```
2018-11-16       1.0  212.457661  189.603541           0.0
2018-11-19       1.0  211.832831  189.759619           0.0
2018-11-20       1.0  210.924644  189.840216           0.0
2018-11-21       1.0  210.067101  189.936764           0.0
2018-11-23       1.0  209.016235  190.032500           0.0
2018-11-26       1.0  208.061743  190.130503           0.0
2018-11-27       1.0  207.216790  190.195936           0.0
2018-11-28       1.0  206.496784  190.286429           0.0
2018-11-29       1.0  205.746838  190.355311           0.0
2018-11-30       1.0  204.945184  190.392028           0.0
2018-12-03       1.0  204.313018  190.462213           0.0
2018-12-04       1.0  203.459201  190.495175           0.0
2018-12-06       1.0  202.539077  190.522235           0.0
2018-12-07       1.0  201.530953  190.511637           0.0
2018-12-10       1.0  200.455751  190.491888           0.0
2018-12-11       1.0  199.345925  190.450452           0.0
2018-12-12       1.0  198.215513  190.414157           0.0
2018-12-13       1.0  197.081890  190.388291           0.0
2018-12-14       1.0  195.785703  190.350684           0.0
2018-12-17       1.0  194.539285  190.299594           0.0
2018-12-18       1.0  193.407487  190.256024           0.0
2018-12-19       1.0  192.183834  190.187669           0.0
2018-12-20       1.0  190.819312  190.107305           0.0
2018-12-21       0.0  189.540965  189.987586          -1.0
2018-12-24       0.0  188.223276  189.833842           0.0
2018-12-26       0.0  186.958919  189.722554           0.0
2018-12-27       0.0  185.767732  189.614768           0.0
2018-12-28       0.0  184.484067  189.514833           0.0
2018-12-31       0.0  183.249006  189.421312           0.0

[1059 rows x 4 columns]
```

[15]:

```
fig = plt.figure(figsize=(14,8))

# Add a subplot and label for y-axis
ax = fig.add_subplot(111,  ylabel='Price in $')

# Plot the Adj Close price
df['Adj Close'].plot(ax=ax, color='b', lw=2.)

# Plot the short and long moving averages
signals[['Short_MA', 'Long_MA']].plot(ax=ax, lw=2.)

# Plot the Signal of Golden Cross
ax.plot(signals.loc[signals.intersection == 1.0].index,
         signals.Short_MA[signals.intersection == 1.0],
         'o', markersize=10, color='gold', label='Golden Cross')
```

```python
# Plot the Signal of Death Cross
ax.plot(signals.loc[signals.intersection == -1.0].index,
         signals.Short_MA[signals.intersection == -1.0],
         'o', markersize=10, color='black', label='Death Cross')

ax.set_title('Golden Cross and Death Cross')
ax.legend(loc='best')

# Show the plot
plt.show()
```


Golden Cross and Death Cross
Adj Close
Short_MA
Long_MA
Golden Cross
Death Cross
Price in $
Date