任务分析

- 使用微软 14-18 年股票数据集训练模型
- 模型 对比，调参 选择最优模型
- 使用模型预测 19-22 年收盘价
- 对预测结果进行可视化
- 分析预测结果

依赖导入

In [5]:

```
from IPython.core.interactiveshell import InteractiveShell
InteractiveShell.ast_node_interactivity = "all"
import matplotlib.pyplot as plt
import numpy as np
from sklearn import linear_model,model_selection
import pandas as pd
from sklearn.model_selection import GridSearchCV
```

数据导入

In [6]:

```
data = pd.read_csv('ms_14-18.csv', delimiter=',')
data.head()
```

Out[6]:

	Date	High Price	Low Price	Open Price	Close Price	Volume
0	2014-09-09 00:00:00	46.970001	46.419998	46.470001	46.759998	40302400.0
1	2014-09-10 00:00:00	46.939999	46.279999	46.820000	46.840000	27302400.0
2	2014-09-11 00:00:00	47.000000	46.470001	46.740002	47.000000	29216400.0
3	2014-09-12 00:00:00	47.020000	46.599998	46.910000	46.700001	38244700.0
4	2014-09-15 00:00:00	46.709999	46.099998	46.540001	46.240002	37667600.0

数据探索

- Date 日期
- High Price 当日最高价
- Low Price 当日最低价
- Open Price 开盘价
- Close Price 收盘价
- Volume 成交量

特征工程

- 特征选择

In [7]:

```
features = [ "High Price", "Low Price" , "Open Price" ,"Volume"]
```

- 划分特征数据集为训练集和测试集

In [8]:
```
X = np.array(data[features])
y = np.array(data["Close Price"])
X_train,X_test,y_train,y_test = model_selection.train_test_split(X,y,test_size = 0.1)
```

建立模型

- 创建 线性回归，岭回归，逻辑回归 模型
- 用训练集训练模型

In [9]:
```
lin_reg = linear_model.LinearRegression()
lin_reg.fit(X_train, y_train)

rid_reg = linear_model.Ridge()
rid_reg.fit(X_train, y_train)

log_reg = linear_model.LogisticRegression()
log_reg.fit(X_train, y_train.astype('int'))
```

Out[9]:
```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=1, normalize=False)
```

Out[9]:
```
Ridge(alpha=1.0, copy_X=True, fit_intercept=True, max_iter=None,
   normalize=False, random_state=None, solver='auto', tol=0.001)
```

Out[9]:
```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
          intercept_scaling=1, max_iter=100, multi_class='ovr', n_jobs=1,
          penalty='l2', random_state=None, solver='liblinear', tol=0.0001,
          verbose=0, warm_start=False)
```

- 选择最优模型

In [10]:
```
print('线性回归 准确率：', lin_reg.score(X_test, y_test))
print('岭回归 准确率：', rid_reg.score(X_test, y_test))
print('逻辑回归 准确率：', log_reg.score(X_test, y_test.astype('int')))
```

```
线性回归 准确率： 0.9997195697970669
岭回归 准确率： 0.9997205693545149
逻辑回归 准确率： 0.06930693069306931
```

In [11]:
```
best_model = rid_reg
```

模型调参

- 使用 网格搜索 对 岭回归 进行调参

In [12]:
```
param={'alpha':[1e-15,1e-10,1e-8,1e-3,1e-2,1,5,10,20,30,35,40,45,50,55,100]}

grid = GridSearchCV(best_model, param, cv=5)

best_model = grid.fit(X_train, y_train).best_estimator_

grid.fit(X_train, y_train).best_estimator_
```

Out[12]:
```
Ridge(alpha=1e-15, copy_X=True, fit_intercept=True, max_iter=None,
   normalize=False, random_state=None, solver='auto', tol=0.001)
```

模型使用

- 载入 19 - 22 年股票数据选取 特征数据集
- 使用 最佳模型 对特征数据集进行预测

In [13]:
```
data = pd.read_csv('ms_19-22.csv', delimiter=',')
data.head()
features = [ "High", "Low", "Open", "Volume"]
X = np.array(data[features])
y = np.array(data["Close"])

pred_y = best_model.predict(X)
```

Out[13]:

	Date	Open	High	Low	Close	Adj Close	Volume	Stock	SMA_50	T_SMA_5(
0	2019-06-03	123.849998	124.370003	119.010002	119.839996	115.725883	37983600	MSFT	NaN	False
1	2019-06-04	121.279999	123.279999	120.650002	123.160004	118.931908	29382600	MSFT	NaN	False
2	2019-06-05	124.949997	125.870003	124.209999	125.830002	121.510231	24926100	MSFT	NaN	False
3	2019-06-06	126.440002	127.970001	125.599998	127.820000	123.431931	21459000	MSFT	NaN	False
4	2019-06-07	129.190002	132.250000	128.259995	131.399994	126.889038	33885600	MSFT	NaN	False

5 rows × 29 columns

结果可视化

- 定义预测准确率函数

In [14]:
```python
# 预测结果与实际值误差 < 0.01 认为准确
def judge(predict, target):
    diff = abs(predict - target) / target
    if diff < 0.01:
        return True
    else:
        return False

# 获取准确率
def get_accuracy(pred_y, y_test):
    size = len(y_test)
    count = 0
    for i in range(0, size):
        if judge(pred_y[i], y_test[i]):
            count += 1
    return count / size
```

- **预测得到准确率**

In [15]:
```python
print('预测准确率为：', get_accuracy(pred_y, y))
```

预测准确率为： 0.8930602957906713

- **拟合结果可视化**

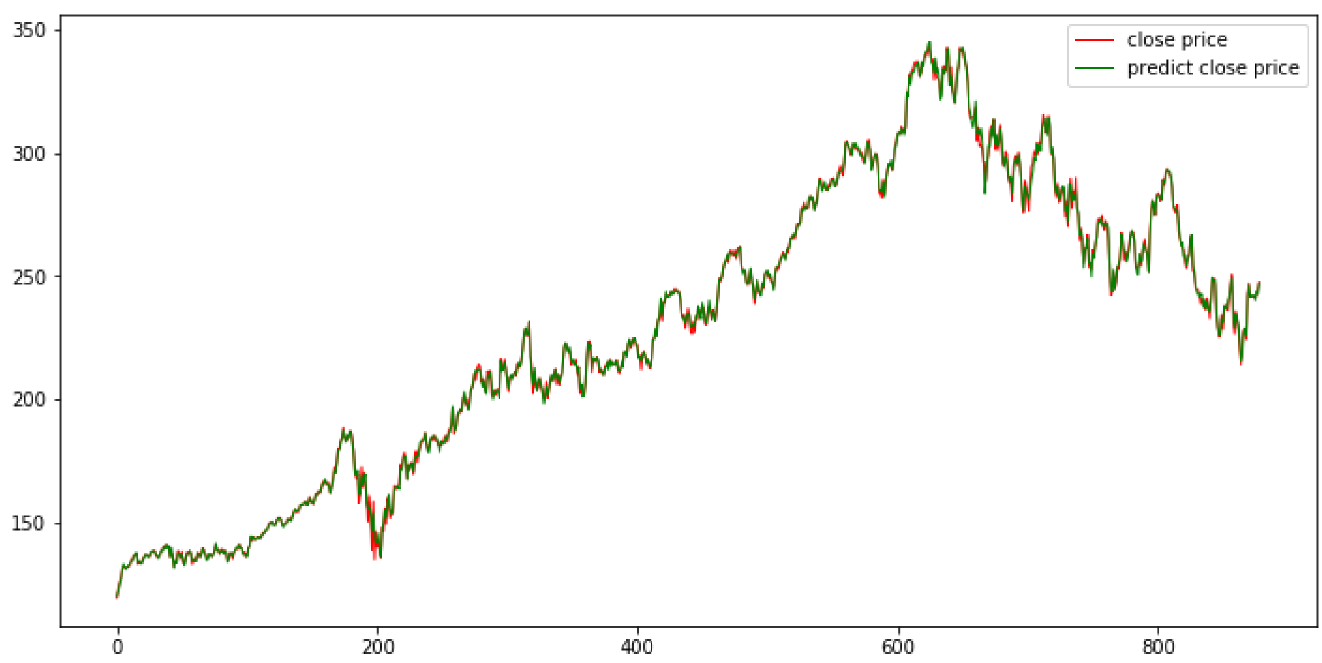
In [29]:
```python
x_axis = [i for i in range(len(y))]
plt.figure(figsize=(12, 6))
plt.plot(x_axis, y, color='red', linewidth=1, label='close price')
plt.plot(x_axis, pred_y, color='green', linewidth=1, label='predict close price')
plt.legend(loc='upper right')
plt.show()
```

Out[29]: <Figure size 864x432 with 0 Axes>

Out[29]: [<matplotlib.lines.Line2D at 0x1f1393216a0>]

Out[29]: [<matplotlib.lines.Line2D at 0x1f1392e54a8>]

Out[29]: <matplotlib.legend.Legend at 0x1f139336198>



结论分析

- 虽然预测的结果误差不大，但是无法用于实际
- 因为股价影响因素还有 经济周期，政策，情绪 等