机器学习第五次作业

1. **模块调用说明**

```
# 库
import pandas as pd
import numpy as np

# 具体函数
from sklearn.model_selection import train_test_split
from sklearn.model_selection import LeavePOut
from sklearn.metrics import mean_squared_error

# 可视化
import matplotlib.pyplot as plt
```

2. **读入数据集**

```
auto = pd.read_csv("C:/Users/mi/Desktop/Auto.csv")
```

---

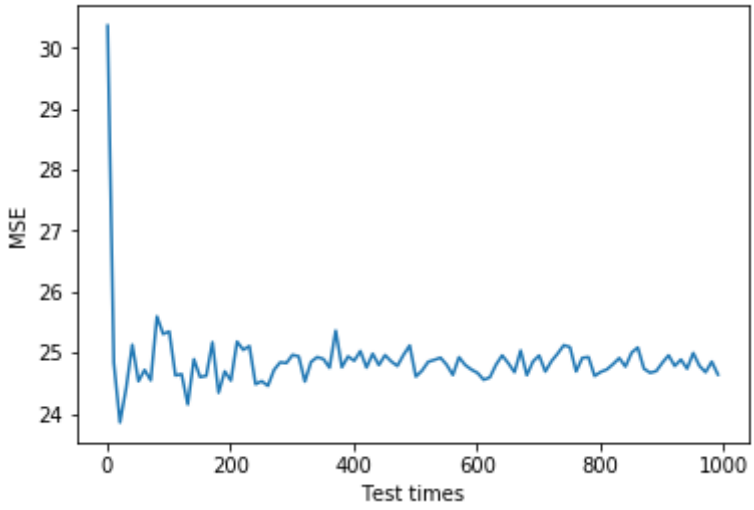
a. 采用多次留出法估计泛化误差

```
# 函数定义
def Pareto_Hold_Out(dataset, times):
    # 测试集占数据集的20%
    # dataset 数据集
    # times 留出法划分次数

    X = auto["horsepower"]
    y = auto["mpg"]
    errors = []
    for i in range(times):
        X_train, X_test, y_train, y_test = train_test_split(X, y, test_size =
0.2)
        y_pred = 40 - 0.15 * X_test
        error = mean_squared_error(y_test, y_pred)
        errors.append(error)

    return np.mean(errors)
```

```
# 调用1
Pareto_Hold_Out(auto, 20)
```

25.12857594936709

```
# 调用2
# 平均泛化误差与试验次数的关系
Times = np.arange(1, 1000 ,10)
Errors = [Pareto_Hold_Out(auto, _) for _ in Times]

plt.plot(Times, Errors)
plt.xlabel("Test times")
plt.ylabel("MSE")
plt.show()
```



b. 采用留p交叉验证法估计泛化误差

```
# b1. K = C(p,N) 时间复杂度过高
def Leave_P_Out(dataset, p):
    # 测试集留出量默认p = 10

    X = auto["horsepower"]
    y = auto["mpg"]
    errors = []
    lpo = LeavePOut(p)
    lpo.get_n_splits(X)

    for train_index, test_index in lpo.split(X):
        X_train, X_test = X[train_index], X[test_index]
        y_train, y_test = y[train_index], y[test_index]
        y_pred = 40 - 0.15 * X_test
        error = mean_squared_error(y_test, y_pred)
        errors.append(error)

    return np.mean(errors)

### b1. K = 20
def Leave_P_Out(dataset, p, K):
    # 测试集留出量默认p = 10
    # 试验次数减少为K次，默认K = 20

    X = auto["horsepower"]
    y = auto["mpg"]
    errors = []
```

```
    for i in range(K):
        X_test = X.sample(n = p, replace = False, axis = 0, random_state = i)
        y_test = y.sample(n = p, replace = False, axis = 0, random_state = i)
        y_pred = 40 - 0.15 * X_test
        error = mean_squared_error(y_test, y_pred)
        errors.append(error)

    return np.mean(errors)
```

```
# 调用
Leave_P_Out(auto, 10, 20)
```

25.648337500000004

---

c.

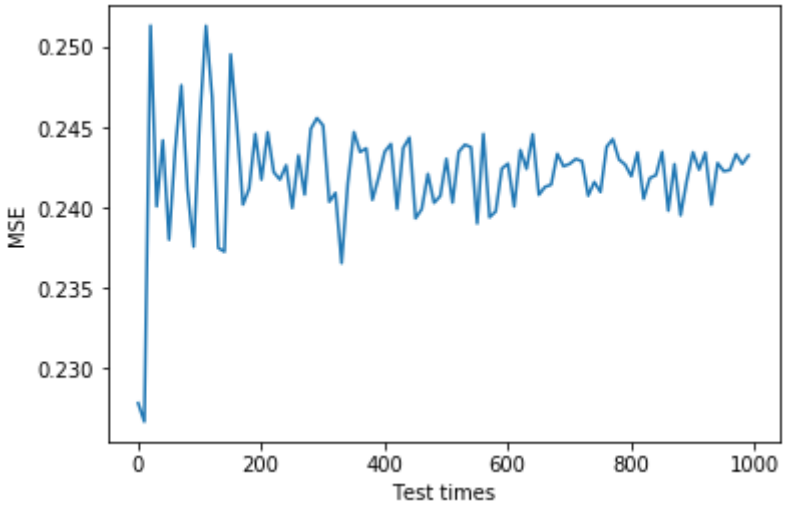
```
# 为auto数据集增加新变量mpg01
auto["mpg01"] = (auto["mpg"] > auto["mpg"].quantile(0.75)).astype(int)
```

```
# 函数定义
def Pareto_Hold_Out(dataset, times):
    # 测试集占数据集的20%
    # dataset 数据集
    # times 留出法划分次数

    X = auto["weight"]
    y = auto["mpg01"]
    errors = []
    for i in range(times):
        X_train, X_test, y_train, y_test = train_test_split(X, y, test_size =
0.2)
        y_pred = (np.e**(3.85 - 0.01 * X_test)/(1 + np.e**(3.85 - 0.01 *
X_test)) > 0.5).astype(int)
        error = mean_squared_error(y_test, y_pred)
        errors.append(error)

    return np.mean(errors)
```

```
# 调用1
Pareto_Hold_Out(auto, 20)
```

25.12857594936709

```
# 调用2
Times = np.arange(1, 1000 ,10)
Errors = [Pareto_Hold_Out(auto, _) for _ in Times]

plt.plot(Times, Errors)
plt.xlabel("Test times")
plt.ylabel("MSE")
plt.show()
```



---

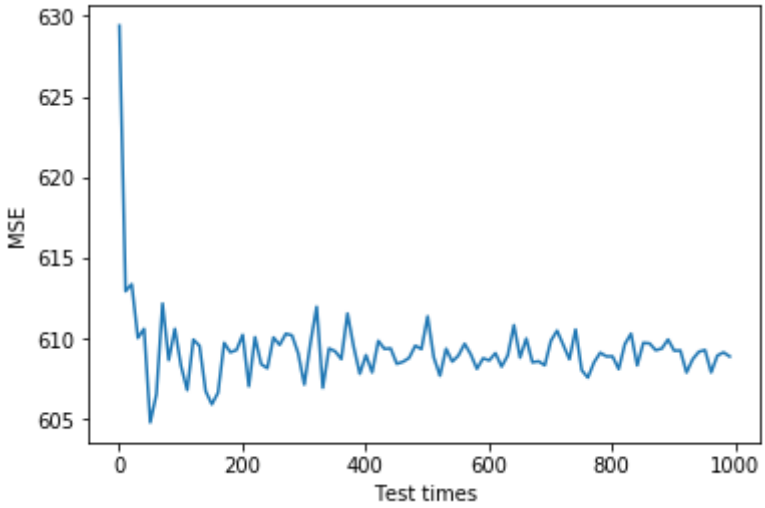
d. 在c的基础上使测试集与训练集内部的不同样例比例一致

```
# 函数定义
def Pareto_Hold_Out(dataset, times):
    # 测试集占数据集的20%
    # dataset 数据集
    # times 留出法划分次数

    X0 = auto.loc[auto["mpg01"] == 0]["weight"]
    X1 = auto.loc[auto["mpg01"] == 1]["weight"]
    y0 = auto.loc[auto["mpg01"] == 0]["mpg"]
    y1 = auto.loc[auto["mpg01"] == 1]["mpg"]
    errors = []
    for i in range(times):
        X0_train, X0_test, y0_train, y0_test = train_test_split(X0, y0,
test_size = 0.2)
        X1_train, X1_test, y1_train, y1_test = train_test_split(X1, y1,
test_size = 0.2)
        X_test = X0_test.append(X1_test)
        y_test = y0_test.append(y1_test)
        y_pred = (np.e**(3.85 - 0.01 * X_test)/(1 + np.e**(3.85 - 0.01 *
X_test)) > 0.5).astype(int)
        error = mean_squared_error(y_test, y_pred)
        errors.append(error)

    return np.mean(errors)
```

```
# 调用1
Pareto_Hold_Out(auto, 20)
```

607.5703417721519

```
# 调用2
Times = np.arange(1, 1000 ,10)
Errors = [Pareto_Hold_Out(auto, _) for _ in Times]

plt.plot(Times, Errors)
plt.xlabel("Test times")
plt.ylabel("MSE")
plt.show()
```



---

e. 抽样方式对泛化误差的影响

根据(a)(b)两问中函数的调用结果，发现不同的抽样方式（留出法与留p法）对回归问题的抽样误差没有显著的影响。

根据(c)(d)两问中函数的调用结果，发现测试集与训练集中样例的一致性（分层等比例）对分类问题的抽样误差有显著的影响。按照分层等比例划分的测试集得到的泛化误差远大于未分层的测试得到的泛化误差。

---

f.习题2.1

为保证训练集与测试集的一致性，在500个正例中选择70%作为训练集的一部分，再在500个反例中选择70%作为训练集的另一部分，余下部分划入测试集。共有$C_{500}^{350} * C_{500}^{350}$种划分凡方式。