西京学院
XIJING UNIVERSITY

# 《机器学习》课程
# 实验操作（Python 版）

# 实验一　初识机器学习——运行环境设置及数据准备

## 实验内容及具体操作方法

1.机器学习常用开发环境的安装、配置（Python）

1.1 下载并安装 Anaconda、Pycharm，并对其进行配置，安装 Numpy、Pandas、Matplotlib、scikit-learn 类库，并在 pycharm 或者 Jupyter Notebook 中检查版本号。

打开 numpy、pandas、matplotlib、scikit-learn 官网，探索相关功能。

```
import numpy as np
import pandas as pd
import matplotlib
import sklearn
print(np.__version__)
print(pandas.__version__)
print(matplotlib.__version__)
print(sklearn.__version__)
```

1.2 掌握使用 Jupyter Notebook 的方法。

添加代码自动补全功能：

命令行运行

```
pip install jupyter_contrib_nbextensions
```

关闭 JupyterNotebook，运行

```
jupyter contrib nbextension install --user --skip-running-check
```

打开 JupyterNotebook，点开 Nbextensions 的选项，并勾选 Hinterland。

2.　学习到 UCI、Kaggle、天池等网站下载鸢尾花数据集、波士顿房价数据集、泰坦尼克号数据集、MNIST 数据集等的方法

下载的数据集，存放在项目目录中。
JupyterNotebook 中，数据集需要导入到当前目录下。

3.对给定的鸢尾花数据集进行加载、查看、图表化显示、预处理，了解算法性能评估过程。

3.1 导入类库

```
from pandas import read_csv  #使用 pandas 来导入数据和对数据进行描述性统计分
```

析

```
from matplotlib import pyplot  #使用matplotlib进行绘图，数据可视化
```

3.2 导入数据集

```
方法1：
from  sklearn import datasets  #使用sklearn自带的示例数据集
iris=datasets.load_iris()
iris.data
iris.target
方法2：
filename='iris.data'    #使用pandas导入数据集
names = ["sepal-length", 'sepal-width','petal-length', 'petal-width', 'class']
dataset=read_csv(filename,names=names)
```

3.3 数据概览

```
#显示数据维度
print('数据维度：行 %s，列 %s' % dataset.shape)
# 查看数据的前10行
print(dataset.head(10))
# 统计描述数据信息
print(dataset.describe())
# 分类分布情况
print(dataset.groupby('class').size())
```

3.4 数据可视化

```
# 箱线图
dataset.plot(kind='box',    subplots=True,    layout=(2,2),    sharex=False,
sharey=False)
pyplot.show()
# 直方图
dataset.hist()
pyplot.show()
# 散点矩阵图
from pandas.plotting import scatter_matrix
```

```
scatter_matrix(dataset)
pyplot.show()
```

3.5 分离数据集

```
from sklearn.model_selection import train_test_split
# train_test_split 函数用于将矩阵随机划分为训练子集和测试子集，并返回划分好的训练集测试集样本和训练集测试集标签。
array = dataset.values
X = array[:, 0:4]
Y = array[:, 4]
validation_size = 0.2   #分离数据，80%训练数据集，20%评估数据集
seed = 7
X_train, X_validation, Y_train, Y_validation = \
train_test_split(X, Y, test_size=validation_size, random_state=seed)
```

3.6 使用逻辑回归和贝叶斯分类器两种算法进行算法性能评估，并在测试集上进行验证。

```
from sklearn.linear_model import LogisticRegression  #逻辑回归
from sklearn.metrics import accuracy_score
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
from sklearn.naive_bayes import GaussianNB  #朴素贝叶斯
from sklearn.model_selection import KFold
from sklearn.model_selection import cross_val_score

# 算法审查
models = {}
models['LR'] = LogisticRegression()
#models['LDA'] = LinearDiscriminantAnalysis()
#models['KNN'] = KNeighborsClassifier()
#models['CART'] = DecisionTreeClassifier()
models['NB'] = GaussianNB()
#models['SVM'] = SVC()
# 评估算法，使用十折交叉验证法
results = []
```

```
for key in models:
    kfold = KFold(n_splits=10, random_state=seed)
    cv_results = cross_val_score(models[key], X_train, Y_train, cv=kfold,
scoring='accuracy')
    results.append(cv_results)
    print('%s: %f (%f)' %(key, cv_results.mean(), cv_results.std()))

    #测试集上验证、比较两种算法分类效果
    lr=LogisticRegression()
    lr.fit(X=X_train, y=Y_train)
    predictions = lr.predict(X_validation)
    print(accuracy_score(Y_validation, predictions)) #预测准确率
    print(confusion_matrix(Y_validation, predictions)) #冲突矩阵
    print(classification_report(Y_validation, predictions)) #数据报告

    nb = GaussianNB()
    nb.fit(X=X_train, y=Y_train)
    predictions = nb.predict(X_validation)
    print(accuracy_score(Y_validation, predictions))
    print(confusion_matrix(Y_validation, predictions))
    print(classification_report(Y_validation, predictions))
```