華東理工大學

模式识别大作业

题　　目　每小时的工资预测

学　　院　信息科学与工程

专　　业　控制科学与工程

组　　员　郑守银

指导教师　赵海涛

完成日期：　2019 年 12 月 2 日

模式识别作业报告——每小时的工资预测

郑宇银

Y30190770

摘要：工资对于人力资源的分配和使用、提高工人劳动效率与社会稳定有着重要意义。从一堆数据集中对工人的工资进行预测，将机器学习中的支持向量机和随机森林算法应用于工人每小时的工资预测中，通过对数据集的预处理、特征提取、变量关联性分析，建模训练以及参数调优，有效地预测出工人每小时的工资。

关键词：工资预测；支持向量机；随机森林；建模

1 引言

工资是雇主对员工的劳动所支付的报酬，工资是劳动者劳动收入的主要组成部分，是衡量收入、分配与劳动力发展水平的重要指标[2]。有效进行工资地预测与分配，有利于人力资源的合理分配和使用，有效地提高工人劳动效率，还关系着社会的稳定。

应用机器学习来预测工资，在大量数据集中，探寻工资与各个变量之间的关系，能有效地预测工资，给企业带来了极大的便利。

2 整体解决方案

2.1 问题分析

搜集工人每小时工资与工人年龄、受教育年数和性别等变量，对工人每小时工资的进行预测。

数据集共有 2 个，分别是 Income_training.csv、Income_testing.csv，数据集中包含 3 个特征、1 个目标变量。

表 1 每小时的工资预测数据集的变量说明

变量类型	变量名	变量含义
特征变量	age	员工的年龄
	yearsEducation	受教育的年数
	sex1M0F	性别（男性 1，女性 0）
目标变量	compositeHourlyWages	员工每小时工资的加权平均值

该项目的评价标准为均方根误差 RMSE，即

$$RMSE = \sqrt{\frac{1}{\mathrm{m}}\sum_{i=1}^{m}(yf^{(i)} - y^{(i)})^{2}}$$

RMSE 越小，说明模型预测的越准确。

2.2 数据预处理

2.2.1 数据加载与预览

（1）导入数据

```
import pandas as pd                                  #导入 pandas 库
import numpy as np                                   #导入 numpy 库
from pandas import Series,DataFrame                  #导入 Series,DataFrame 模块
import matplotlib.pyplot as plt                      #导入 matplotlib 库
df=pd.read_csv('Income_training.csv')                #读取 CSV 数据
df.head()                                            #展现前五行数据
```

	compositeHourlyWages	age	yearsEducation	sex1M0F
0	21.38	58	10	1
1	25.15	42	16	1
2	8.57	31	12	0
3	12.07	43	13	0
4	10.97	46	12	0

图 1 每小时工资预测数据集的前 5 行数据

图 1 中展现了 Income_training.csv 数据集中前 5 行的数据。可以看到，sex1M0F 为离散型变量的数据，其余均为连续型变量。

（2）数据预览

```
print(df.info())                                     #快速查看数据的描述
```

输出结果：

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3197 entries, 0 to 3196
Data columns (total 4 columns):
compositeHourlyWages      3197 non-null float64
age                       3197 non-null int64
yearsEducation            3197 non-null int64
sex1M0F                   3197 non-null int64
dtypes: float64(1), int64(3)
memory usage: 100.0 KB
```

```
None
```

可以看到数据集的样本量有 3197 个，变量有 4 个，不存在缺失值的变量；age、yearsEducation、sex1M0F 为 int 类型变量，compositeHourlyWages 为 float 类型变量，不需要数据转换。

（3）数据统计信息预览

```
df.describe()                                    #数据的描述性统计信息
```

输出结果如图 2 所示。

	compositeHourlyWages	age	yearsEducation	sex1M0F
count	3197.000000	3197.000000	3197.000000	3197.000000
mean	15.495127	36.884579	13.180794	0.491711
std	7.754763	11.996980	3.042127	0.500009
min	2.300000	16.000000	0.000000	0.000000
25%	9.250000	28.000000	12.000000	0.000000
50%	14.210000	36.000000	13.000000	0.000000
75%	19.650000	46.000000	15.000000	1.000000
max	49.920000	65.000000	20.000000	1.000000

图 2 每小时工资预测数据集的描述性统计信息

通过各变量的个数 count 也可以看出，各变量不存在缺失值。

2.3 特征提取

对数据集进行特征提取，一方面观察每个变量的数据分布情况，另一方面筛选与目标变量关系显著特征、摒弃非显著特征，有助于提高算法的效果和性能。

2.3.1 变量特征图表

对变量作直方图、箱线图等可视化分析，观察各变量的样本分布情况。

（1）变量直方图

```
df.hist(xlabelsize=8,ylabelsize=8,layout=(2,2),figsize=(10,8))
                                #绘制直方图，labelsize-标签大小，layout-布局，figsize-图形大小
plt.show()
```

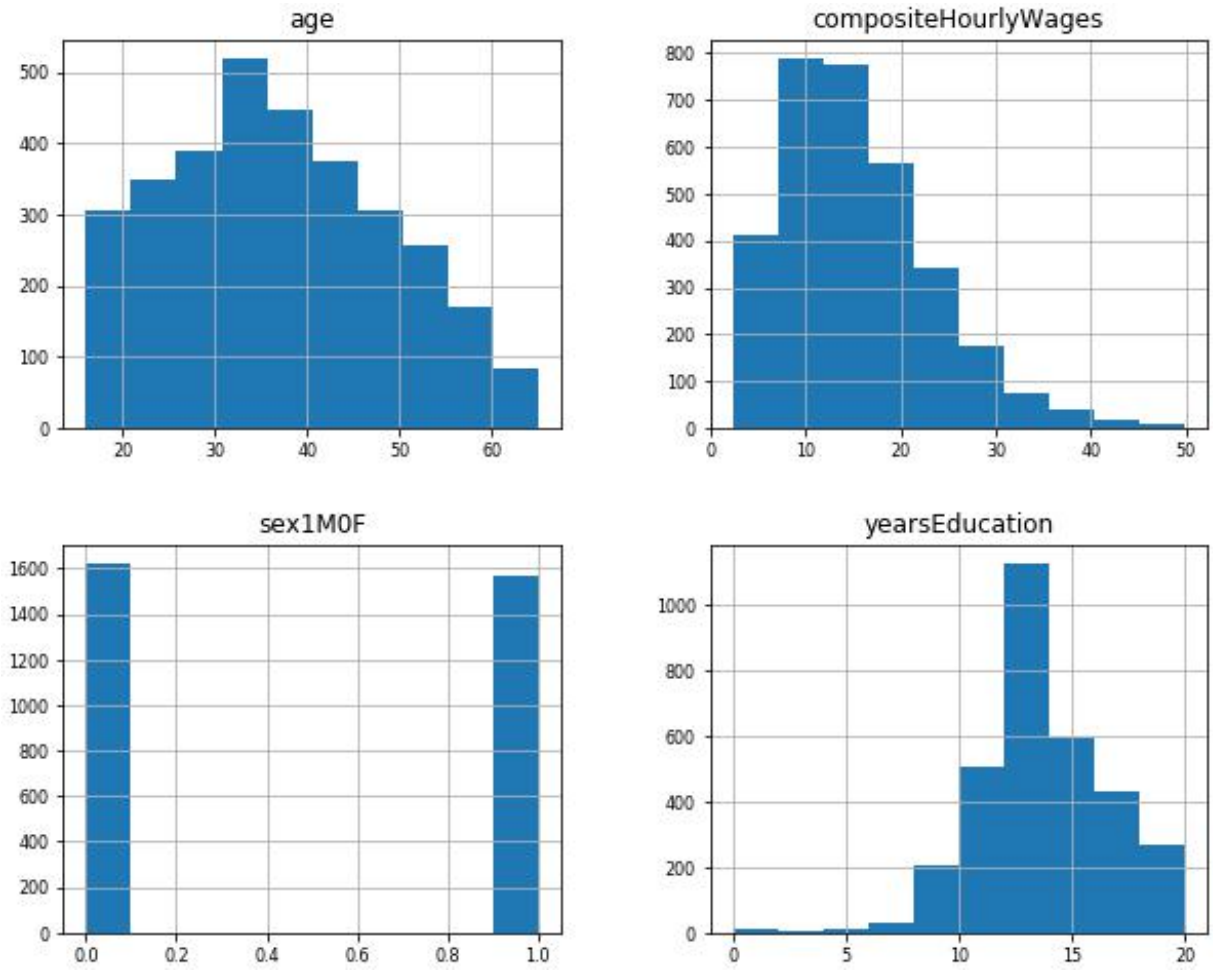

图 3 每小时工资预测数据集的变量直方图

（2）变量箱线图

```
df.plot(kind='box',subplots=True,layout=(2,2),sharex=False,sharey=False,fontsize=12,figsize=(20,12))
                                    #绘制箱线图,subplot-子图,share 共用 y 轴，fontsize 字体大小
plt.show()
```

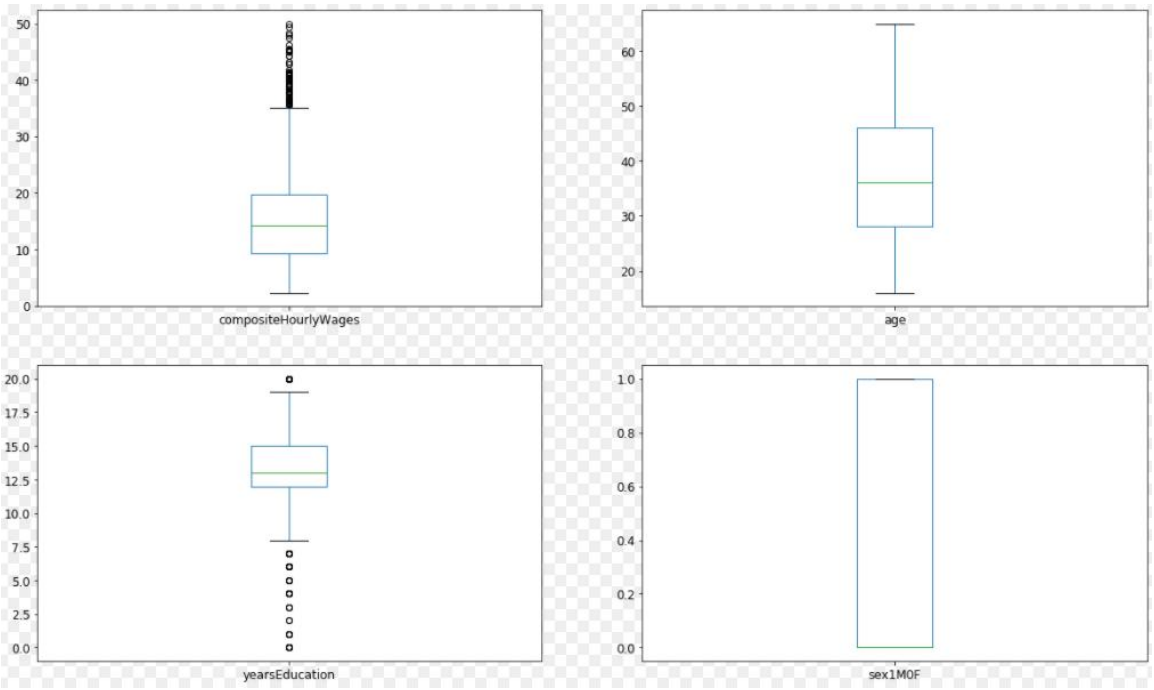

图 4 每小时工资预测数据集的变量箱线图

2.3.2 变量关联性分析

接下来观察各特征变量与目标变量的关联情况，从而更好地筛选显著特征变量。

（1）与目标变量的相关系数

```
corr_matrix=df.corr()                                    #生成相关系数矩阵
print(corr_matrix['compositeHourlyWages'].sort_values(ascending=False))
                                    #打印 compositeHourlyWages 与其他变量的相关系数
```

提取 compositeHourlyWages 的相关系数，并按降序排序，输出结果为：

```
compositeHourlyWages    1.000000
age                     0.361892
yearsEducation          0.303610
sex1M0F                 0.209863
Name: compositeHourlyWages, dtype: float64
```

可以看到，与目标变量 compositeHourlyWages 相关系数最大的为 age，最小的是 sex1M0F，但各个相关系数差距并没有很大。

（2）目标变量与特征向量的散点图

```
plt.figure(figsize=(8,10))                               #设置图形大小
plt.subplot(311)                                         #3 行 1 列，第 1 个图
```

```
plt.scatter(df['compositeHourlyWages'],df['age'])
                                        #绘制 compositeHourlyWages 与 age 的散点图


plt.subplot(312)                                        #3 行 1 列，第 2 个图
plt.scatter(df['compositeHourlyWages'],df['yearsEducation'])
                                #绘制 compositeHourlyWages 与 yearsEducation 的散点图
plt.subplot(313)                                        #3 行 1 列，第 3 个图
plt.scatter(df['compositeHourlyWages'],df['sex1M0F'])
                                        #绘制 compositeHourlyWages 与 sex1M0F 的散点图
plt.show()
```

分别绘制 compositeHourlyWages 与 age、compositeHourlyWages 与 yearsEducation、compositeHourlyWages 与 sex1M0F 的散点图，在同一区域展示。

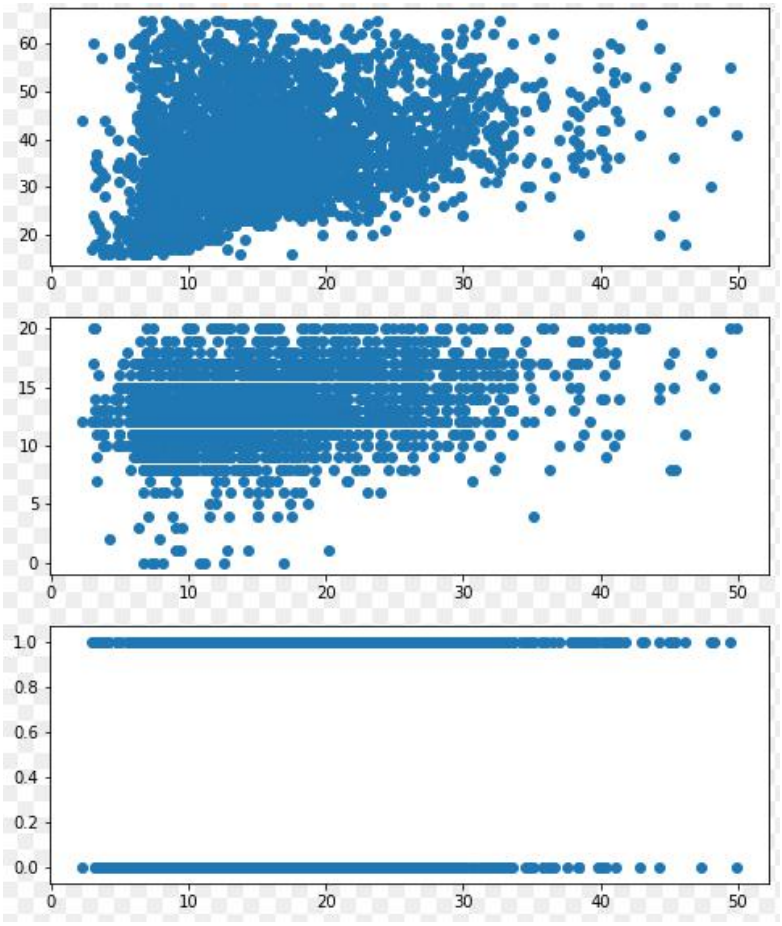

图 5 每小时工资预测数据集的变量散点图

可以看到各特征变量与目标变量直接的线性变化关系 55 开。所以决定不予剔除相关系数较小的特征变量。

2.4 建模训练

2.4.1 对训练数据集的划分

虽然项目中提供了要预测的数据集 Income_testing.csv，但是该数据集中没有目标变量 compositeHourlyWages，因此无法帮助我们对所采用的算法进行效果评估。因此，对 Income_training.csv 数据集进行划分，获取训练基与验证集，从而对所使用的算法进行评估。

选取特征变量并划分数据集

```
col_n=['age','yearsEducation','sex1M0F']                    #设置变量列表
X=df[col_n]                                                  #选取特征变量
y=df.compositeHourlyWages                                    #设定 compositeHourlyWages 为 y

from sklearn.model_selection import train_test_split      #导入数据划分包

#把 x、y 转化为数组形式，以便于计算
X=np.array(X.values)
y=np.array(y.values)
#以 25%的数据构建测试样本，剩余作为训练样本
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.25,random_state=1)
                                        #random_state-随机数种子，为 1，每次随机数一样

X_train.shape,X_test.shape,y_train.shape,y_test.shape
```

特征变量有三个 age、yearsEducation、sex1M0F，数据集划分比例为 25%。

输出结果为：

```
((2397, 3), (800, 3), (2397,), (800,))
```

训练集的样本量有 2397 个，验证集的样本量有 800 个，使用的特征变量有 3 个。

2.4.2 采用不同算法的建模训练

为更好地得到较好的预测结果，选择不同的算法模型并进行训练。

（1）支持向量机回归

```
from sklearn.svm import SVR
                                        #从 sklearn.svm 中导入支持向量机回归模型
linear_svr=SVR(kernel='linear')
                                #使用线性核函数配置的支持向量机进行回归训练并预测
linear_svr.fit(X_train,y_train)                  #训练模型
```

```
y_hat_svr=linear_svr.predict(X_test)          #预测验证集的预测
from sklearn import metrics                   #从 sklearn 中导入 metrics 函数
print ("RMSE_svr:",np.sqrt(metrics.mean_squared_error(y_test,y_hat_svr)))
                                              #用 scikit-learn 计算 RMSE
```

输出结果：

```
RMSE_svr: 6.872665833427825
```

得到均方误差根 RMSE=6.872665833427825

（2）随机森林回归算法

```
from sklearn.ensemble import RandomForestRegressor
                                              #从 sklearn.ensemble 中导入随机森林回归
#使用随机森林进行回归训练并预测
rf=RandomForestRegressor(random_state=200,max_features=0.3)
rf.fit(X_train,y_train)

y_hat_rf=rf.predict(X_test)
from sklearn import metrics
print ("RMSE:",np.sqrt(metrics.mean_squared_error(y_test,y_hat_rf)))
```

输出结果：

```
RMSE: 7.428207940140678
```

得到均方误差根 RMSE=7.428207940140678

经比较可以看出，支持向量机回归模型 SVR 的均方误差根 RMSE 更小，决定采用主持向量机回归。

2.4.3 **参数调优**

支持向量机参数优化

```
alphas_svr=np.linspace(0.1,1.2,20)            #设置惩罚项 C 的等差参数序列
rmse_svr=[]                                   #设置 RMSE 列表
for c in alphas_svr:
    model=SVR(kernel='linear',C=c)            #设定模型为 SVR
    model.fit(X_train,y_train)                #使用训练数据进行参数求解
    y_hat=model.predict(X_test)               #预测划分的训练集
```

```
        rmse_svr.append(np.sqrt(metrics.mean_squared_error(y_test,y_hat)))
                                                    #将得到的均方根误差结果加入 RMSE 列表中

    plt.plot(alphas_svr,rmse_svr)                   #绘制不同 c 取值的 RMSE 结果
    plt.title('Cross Validation Score with Model SVR')   #添加标题
    plt.xlabel("alpha")                             #添加 x 轴标签
    plt.ylabel("rmse")                              #添加 y 轴标签
    plt.show()
```

将惩罚项 C 设置在 0.1 到 1.2 之间的等差序列列表 alphas_svr，数量为 20 个；采用 for 循环方式，将 C 从列表中依次取值，进行 SVR 训练，并得到 RMSE 的值，将获取的不同的 RMSE 值也加入到列表 rmse_svr 中；绘制不同 C 取值下对应的 RMSE 值的折线图。

输出结果：

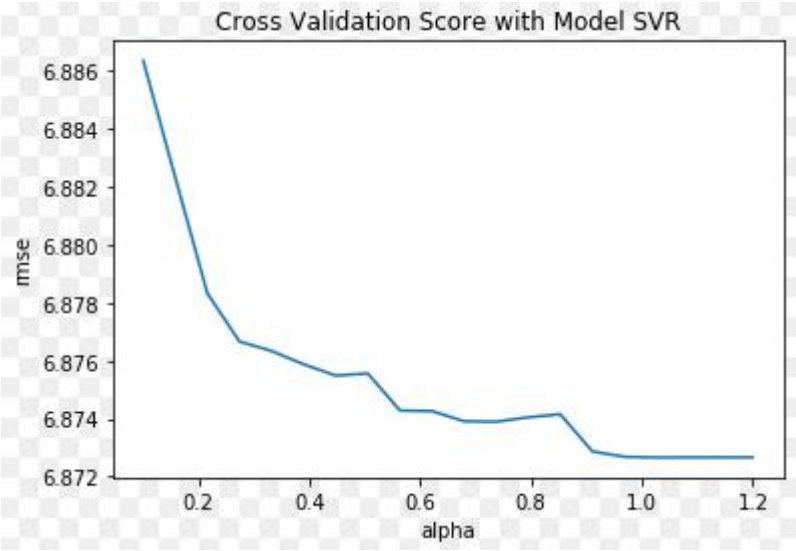

图 6 每小时工资预测数据集的 SVR 交叉验证结果

从图中可以看出，随着惩罚项 C 取值的增大，SVR 算法的 RMSE 越小。但在 C=1.0 处 RMSE 值变化趋缓，基本在 1.0~1.2 之间，能得到最小的 RMSE，其值在 6.872~6.886 之间。所以决定将惩罚项 C 设置为 1.0。

2.5 预测与提交结果

在对训练模型以及最优参数确定之后，对 Income_testing.csv 数据进行预测，并提交预测结果。

（1）预测 Income_testing 数据并提交结果

```
    df_test=pd.read_csv('Income_testing.csv')                    #读取 CSV 数据
```

```
df_test.head()                                    #展示前 5 行数据
```

先对 Income_testing.csv 数据进行加载与预览，如图 7 所示。

	ID	age	yearsEducation	sex1M0F
0	1	36	20	0
1	2	38	17	0
2	3	24	10	0
3	4	39	12	1
4	5	50	12	0

图 7 每小时工资预测数据集的 Income_testing 数据集

可以看到各个变量皆为 int 类型，所以无需对数据进行转换。

（2）预览 Income_testing 数据信息

```
print(df_test.info())                             #快速查看数据的描述
```

对 Income_testing.csv 数据进行查看有无缺失值，输出结果：

```
RMSE: <class 'pandas.core.frame.DataFrame'>
RangeIndex: 800 entries, 0 to 799
Data columns (total 4 columns):
ID                    800 non-null int64
age                   800 non-null int64
yearsEducation        800 non-null int64
sex1M0F               800 non-null int64
dtypes: int64(4)
memory usage: 25.1 KB
None
```

可以看到预测数据 Income_testing.csv 不存在缺失值的情况，所以不需要进行缺失值处理。

（3）Income_testing 数据的预测与结果的提交

```
testX=df_test[col_n]                              #选取特征变量
svr_test=SVR(kernel='linear',C=1.0)
                                  #使用线性核函数配置的支持向量机进行回归训练并预测
svr_test.fit(X_train,y_train)                     #训练模型


testy_svr=svr_test.predict(testX)                 #对测试集的预测
```

```
submit=pd.read_csv('Income_sample_submission.csv')      #读取 csv 的数据
submit['compositeHourlyWages']=testy_svr
submit.head()                                           #展示前 5 行
```

特征变量与训练时的特征变量相同，参数设置 C=1.0，这里是对 Income_training.csv 全部样本数据进行模型训练，然后对测试集 Income_testing 进行预测。

将预测值导入 Income_sample_submission.csv 文档中，同时在该文档中形成新的变量——compositeHourlyWages，即对预测集的数据预测得到的目标变量。前五行数据的展示结果如图 8 所示。

	ID	compositeHourlyWages
0	1	18.921767
1	2	16.836399
2	3	6.962125
3	4	15.734733
4	5	15.686120

图 8 Income_sample_submission 数据预测的前 5 行

3 小组分工

程序设计及编写：郑守银

程序调试：郑守银

报告：郑守银

4 总结

对每小时工资预测数据集进行预处理、特征提取、变量关联性分析，建模训练以及参数调优，模型选择支持向量机和随机森林。

通过仿真可以发现，支持向量机的效果更好，参数调优后惩罚项 C 选择 1.0。在对测试集进行预测，最终得到测试集中工人每小时的工资。

5 致谢

本科对模式识别知之甚少，听了赵老师的授课之后，知道了机器学习是怎么一回事，也懂得了一些算法，但总归是课本上的内容，只知道推公式，但不清楚这些算法如何使用，也不知其效果。

学习的目的在于应用，在于能够用理论去指导实践，实践出真知。

第一次做机器学习的作业，最开始是担心自己做不出来。之后问同学，自己查找资料，从环境到编译器，到库的安装，再到 kaggle 上面找项目，再到完成一个小项目，算是入门了。虽然我只是懂得皮毛，但渐渐地明白了 python 语言如何进行编写，如何应用这些算法解决实际问题。

参加机器学习是一个不错的锻炼自己机器学习知识水平的机会。最后感谢赵老师悉心教导，也感谢帮助过我的同学们。

参考文献：

[1] 张居营.大话python机器学习[M].北京:中国水利水电出版社,2019:385-407

[2] 李媛.多元线性回归在平均工资预测中的应用研究[J].信息通信,2018(01):31-33