Basketball Match Prediction

by Q . J . Y, LOOKCC, Yue Pan

一、运行环境

操作系统和语言

- OS: Archlinux-4.12.13-1
- Python: 3.6.2

依赖库

(以下依赖库名称以 arch 的 pacman 内的为准)

- python-tensorflow-cuda
- python-pandas
- python-numpy
- python-scikit-learn

运行步骤

```
cd DNN
python get_win_rate.py
python dnn.py
```

进入 DNN 文件夹，依次运行 `get_win_rate.py` 和 `dnn.py`，预测的结果将会输出至项目文件夹下的 `result.csv` 中，经过稍加修改为 `predictPro.csv` 文件。

二、基本说明

1. 本次预测采用 DNN(深度神经网络)模型，以之前的比赛数据来构造向量，用作训练数据。
2. API 选用 Tensorflow 框架的 cuda 版本，用 GPU 运行加快计算速度。
3. 在数据的处理上，矩阵处理使用 numpy 库，csv 文件处理使用 pandas 库。
4. 采用 PEP8 标准 python 代码规范，格式清晰，简介明了。

三、API 说明

PCA

函数原型：

```
sklearn.decomposition.PCA(n_components = None, copy = True, whiten = False)
```

参数说明：

- n_components：PCA 算法中所要保留的主成分个数 n，也即保留下来的特征个数 n
- copy：表示是否在运行算法时，将原始训练数据复制一份，缺省时默认为 True。
- whiten：白化，使得每个特征具有相同的方差，缺省时默认为 False。

PCA 对象的属性：

- explained_variance_ratio_：返回所保留的 n 个成分各自的方差百分比。

PCA 对象的方法：

- fit_transform(X)：用 X 来训练 PCA 模型，同时返回降维后的数据。

Tensorflow

函数原型：

```
feature_columns = [tf.feature_column.numeric_column("x", shape = [] )]
```

功能：选取特征值，这里在 `shape = []`里面填上 `len(training_set[0])`，也就是特征值的个数

函数原型：

```
classifier = tf.estimator.DNNClassifier(feature_columns = feature_columns, hidden_units = hidden_units, n_classes = 6, model_dir = "./seedcup_model")
```

功能：构造分类器

参数说明：

- feature_columns：把之前创建的特征列传入。
- hidden_units：每层神经元数量。
- n_classes：目标的类型的个数。
- model_dir：训练模型保存的路径。

函数原型：

```
test_input_fn = tf.estimator.inputs.numpy_input_fn(x, y, num_epochs = 1, shuffle = False)
```

功能：得到数据输入函数

参数说明：

- x, y：训练数据和标记
- shuffle：是否打乱数据

返回值：数据输入函数

函数原型：

```
classifier.train(input_fn = train_input_fn, steps = 38000)
```

功能：进行训练

参数说明：

- input_fn：数据输入函数
- steps：迭代次数

函数原型：

```
classifier.evaluate(input_fn = test_input_fn)
```

功能：评价训练的准确度

参数为数据输入函数，返回值为一个字典，内含 loss, auc, global step,

accuracy 四个数据。

函数原型：

```
classifier.predict(input_fn = predict_input_fn)
```

功能：进行预测，参数为数据输入函数，返回值为预测的结果的 list

四、数据特征值的选取

本次初赛最终决定只使用球队在之前比赛的数据，构造的特征向量为：[客场球队，主场球队，客场球队胜场次数，客场球队负场次数，主场球队胜场次数，主场球队负场次数]。

在之前有尝试过使用球员的数据，比如选取一个球队中所有球员的三个命中率的平均值，选取一个球队最厉害的十个球员的各项数据，选取各项指标的方差等等，但是最终测试起来效果并不高，考虑到比赛本来就是团队之间的竞争。

此前的胜负场次作为团队竞争的结果，本来就能说明一些球员的综合数据。因此只选用球队之间比赛的数据，说服力还是比较强的。

再者鉴于只有七千条数据，不适合选取太复杂的特征值，最终就决定只使用比赛数据。

五、预测模型与优化

我们采用 DNN 进行训练与预测。

起初我们尝试了多种模型，如逻辑回归、线性回归、支持向量机等，效果都不是特别好，经过多次测试，我们最终选择了 DNN 作为最终的模型。我们调用了 tensorflow 的 DNNClassifier 来进行训练和预测，使用起来比较方便且能充分利用特征来进行分类。

虽然本次初赛的训练数据不是很多，但用 DNN 还是可以给出不错的结果，因此我们最终采用了 DNN 进行预测。

整个训练过程中的可调参数为：

- PCA 降低至的维度 n
- 隐藏层
- 训练的迭代次数

测试阶段选用随机的 6000 组数据作为训练数据，剩下的数据来模拟测试。通过脚本遍历参数运行程序，来得到测试结果，最终从所有的数据中选出了准确度最高的提交。

在优化上，我们使用了 PCA 进行降维处理，并且每训练 1000 次打印一次结果，以便观察训练的效果。