1. 数据预处理

从中国票房网、豆瓣影评、百度搜索指数中爬取了近 3 年的电影信息，共两百多部。提取了主要的特征，包括上映年份、片长、豆瓣评分、星数、评价人数、5 星比例等、上映前 15 天的百度搜索指数共 30 维的特征向量，及其目标变量电影票房数（以百万为单位）。为减少各特征分布对结果的影响，对特征变量作标准化

$$\mathcal{X}_{\mathrm{i}} = \frac{\mathcal{X}_{\mathrm{i}} - \mu}{\sqrt{\mathrm{var}}}$$

其中$\mathcal{X}$是某列特征，μ是其均值，var 是方差，$\mathcal{X}_{\mathrm{i}}$是第 i 个分量。

代码在 extract_feats.py 中，处理后的数据在 data.txt 中，最后一列是电影票房。

2. 模型选择

这是一个回归问题，我们选用了 Spark 中 MLlib 里的 LinearRegression 模型，即线性回归模型。

#首先，将每一行的特征向量和目标变量封装成 MLlib 中的 LabeledPoint 类

```
data=records.map(lambda r:LabeledPoint(float(r[-1]),[float(field) for field in r[0:-1]]))
```

#训练模型

```
linear_model = LinearRegressionWithSGD.train(data, iterations=200, step=0.9, intercept=True)
```

#用模型作预测

```
true_vs_predicted = data.map(lambda p: (p.label, linear_model.predict(p.features)))
```

#评价模型

我们用 R^2 系数作为模型的评价指标，R^2 是一个 0-1 之间的数，值越接近 1，表示模型预测能力越好。定义如下

$$\mathrm{R}^2 = 1 - \frac{\sum_{\mathrm{i}}(\mathrm{y_i} - \mathrm{f_i})^2}{\sum_{\mathrm{i}}(\mathrm{y_i} - \bar{\mathrm{y}})^2}$$

其中，y_i 是票房真实值，f_i 是票房预测值，$\bar{y}$是真实值的平均值。

代码在 movie_linear.py 中。

3. 参数调优

（1）迭代次数

我们以 R^2 系数为指标，设置步长为 0.9 时，迭代次数分别取 100，120，130，150，180，190，200，220，观察 R^2 系数的变化如下图 1 所示，横坐标是迭代次数，纵坐标是 R^2 系数，可以看出增加迭代次数，可以提高模型精度，但当迭代次数超过某个值后，模型性能不会再提高。

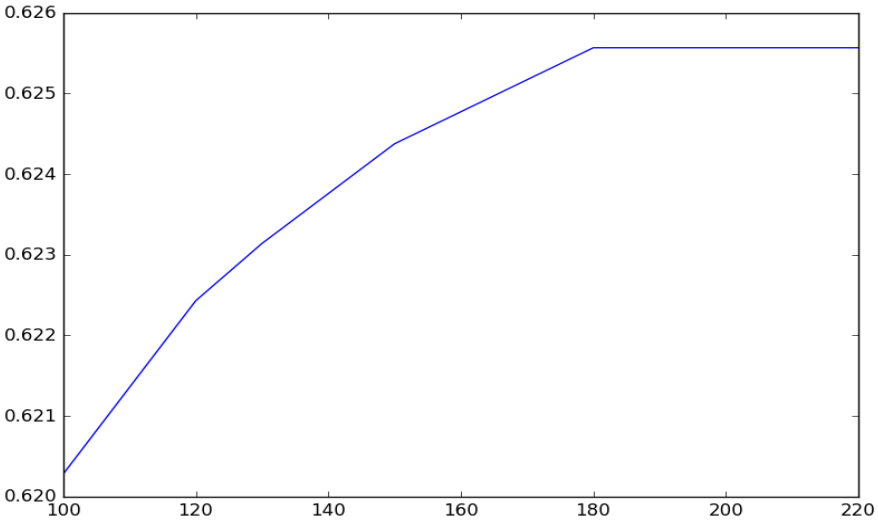

图 1 迭代次数对 R^2 值的影响

（2）步长

设置迭代次数 200，步长分别为 0.6，0.65，0.7，0.75，0.8，0.85，0.9，0.95 时，R2 系数的变化如下图 2 所示，横坐标是迭代次数，纵坐标是 R^2 系数

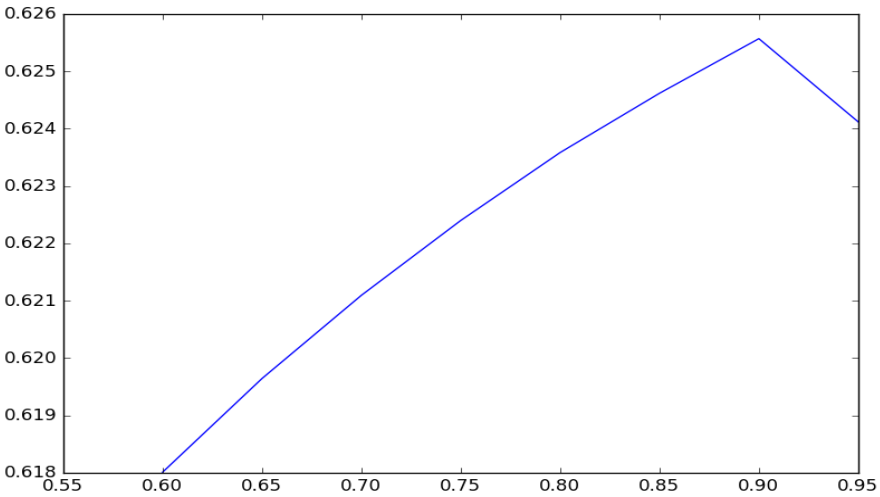

图 2 步长对 R^2 值的影响

4. 结果分析

在综合考虑了豆瓣评分和百度搜索指数后，预测模型的 R^2 系数可以达到 0.63 左右下图展示了两百多部电影的预测票房与真实票房的结果。

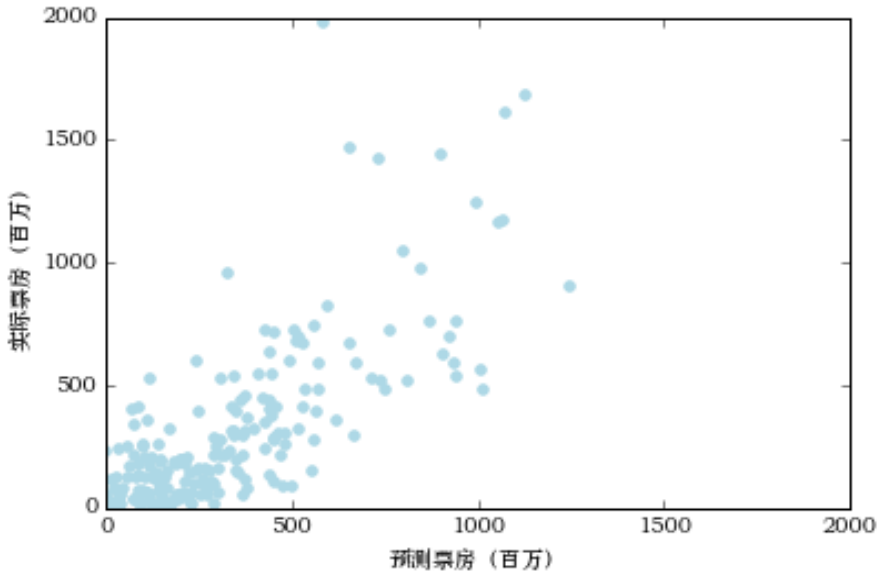

图 3 票房预测值与真实值结果

5. 改进

（1）对于百度搜索指数的特征，直接将上映前 15 天的数据归一化作为特征放入线性模型中，这不能充分分析该时间序列的潜在信息，可以作频域分析或用神经网络等更复杂的模型挖掘该时间序列的信息；

（2）可以加入更多的社交网络的数据，综合考虑更多其他的票房影响因素如电影导演、演员、编剧的社会号召力、制片公司的投资规模和宣传力度、类型、产地、档期等提高预测精度。