价格预测 2

生 52 张斐然 2015012332

1. 预处理方式：

预处理方式在价格预测 1.5（即第一次作业的补充内容）的基础上进行了进一步的修改，具体如下：

1.1 价格计算：

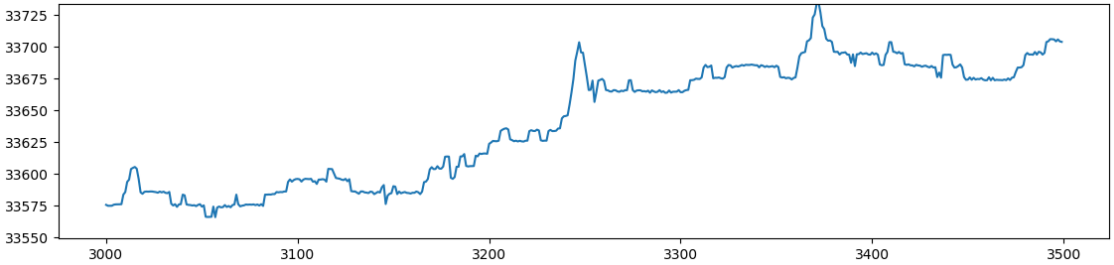
a) 由于在价格预测 1 中对期货数据的理解不够，所以只用到了 lastPrice 的数据，然而由于交易经常每秒有许多次，如果只用其中的两次采样则显然利用的不充分。

b) 因此，我将价格修改为了助教所给的公式，即 k 乘以两次 turnover 差除以两次 volume 差（区间价格），加上 1-k 乘以 bid 与 ask 的平均（最新价格）。

c) 原理分析：前者占比相对更小，因为反映的是 0.5s 区间内的平均信息，而后者占比更大，因为反映的是当下最新的交易信息。

d) 同时，因为每次交易的具体价格是一个离散值，且变化频率很大，考虑区间上的信息便能起到了平滑和降低噪音的作用。此外，由于 lastPrice 往往是 bid 与 ask 其中之一，将其取平均也能起到上述效果。

e) 选择将 k 设为了 0.25，可以看到价格曲线已经很平滑了，所以也无需继续增加 k 值。



1.2 缺失处理：

如果前 0.5s 没有交易，则用 lastPrice 代替区间价格（此时两者实际上一样）；而如果前 0.5s 没有采样数据（行情没有变化），则用 0.5s 前的价格来填充。

1.3 数据标注：

如果考虑每个点之后 10~40 个点（5~20 秒），计算与当前点的价格相差最大的点，并以变化率相对于 0.015%来标注上涨或是下跌，可以得到标注结果如下：

	跌	平	涨
A1	37775	919124	38762
A3	45389	1260816	45461
B2	9544	1217797	8857
B3	15254	1309369	12808

1.4 特征向量：

a) 对于每个点，抽取其之前的 dim 个点（dim 设为 100 左右），减去当前点的价格（中心化），作为 dim 维的特征向量，并以未来价格变化作为标签。

b) 关于交易量，虽然在前期发现它和价格变化的关系很大，往往一笔较大交易的交易就伴随着一个涨或跌，但是深入思考后，发现交易量其实是“果”而非“因”。也即是说，交易量只是在之前价格的变化趋势的作用下，使得当前价格发生变化的机制。所以如果要预测几秒之后的价格变化的话，几秒前的交易量并不能提供有用的信息。当然这还有待经过实验验证，因为交易量的信息的噪音很多，必须很仔细的对其处理后才可能有用。

c) 经过试验后发现，如果直接将 dim 维的价格向量再加上 dim 维的交易量向量，进行预测的准确率平均会下降 5 个百分点。

d) 如果能够利用的话，我认为最重要的是要找到对其归一化的方式才行。

1.5 异常处理：

a) 用正则将“2017/.*2017-”替换为“2017-”。

b) 删去除了主要的四类合约之外的类。

c) 考虑到每次交易刚开始时的时候数据异常值较多，因此舍去了每天白天 9 点前、晚上 21 点前的记录数据。

d) 在 0807 这一天中出现了一大段 askPrice 为 0 的异常情况（其他天这种异常只会出现在 9 点前），为此增加了判断，当 ask 或 bid 为 0 时，将 k 临时设为 1。

1.6 数据不平衡的处理：

a) 在涨跌平三类中，将除了最少的一类外，都降采样至最少的那一类的数量。

b) 后续可以通过 bootstrap 等重采样方法来增加数据利用率。

c) 经实验，重采样可以增加约 6 个百分点的准确率，具体见后。

1.7 训练集、测试集划分：

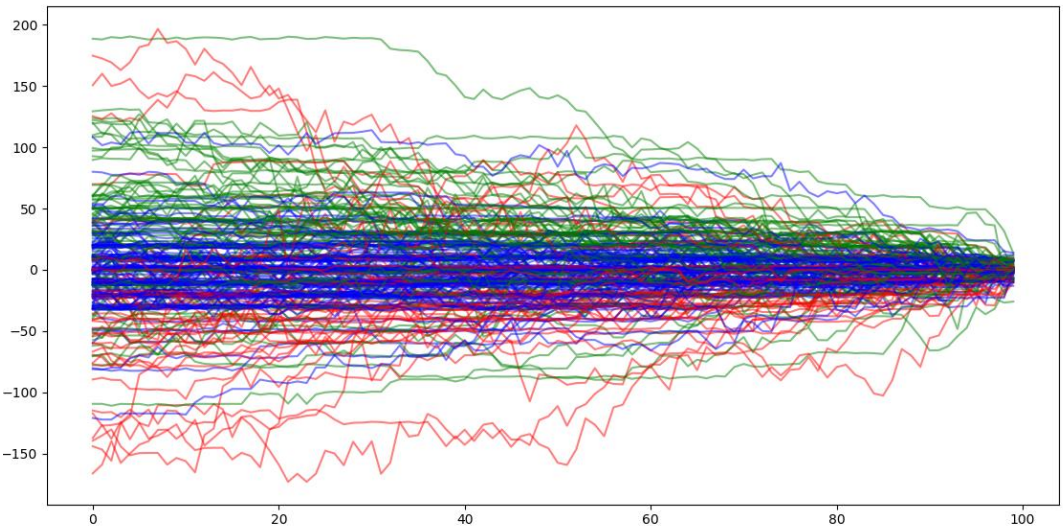
a) 划分方法为，按时间顺序，某个时间点之前的划分为训练集，之后的划分为测试集。

b) 实际划分时，方便起见，我并没有严格按照 8 月 10 号前后来划分，而是通过测试发现该日前后的数据分别占 60%与 40%左右，因此，我直接将经过降采样平衡过后的数据按上述比例来划分。

c) 当然，这样划分可能会导致一些数据的错位，即有可能出现使用 8 月 10 号的数据来预测 8 月 9 号，但是由于随机采样的平均分布，使得其出现的概率很低。

1.8 数据可视化：

a) 将 B2 类平衡后，再次采样每个类的 1/10，画出三个类的大致分布如下：
其中绿色为未来会跌，蓝色为未来持平，红色为未来会涨。



b) 可以看到，预测很大程度上是依照之前的趋势进行的，之前在跌则未来几秒可能继续跌；之前变化不大则未来几秒仍无大变化；之前在涨则未来几秒可能继续涨。

接下来，我使用了三种分类器来对这个几万*100 的矩阵进行分类。首先是 sklearn 中的 neural_network；然后是 TensorFlow 中的 DNNClassifier；最后是 tf 中的 LSTM。

2. 预测结果：

三个分类器中，由于前两者的本质都一样，都是多层神经网络，同时表现也相似，因此主要使用第一个来呈现结果并进一步调参。而 LSTM 由于对 TensorFlow 的使用不够熟练，导致实现的很粗糙，训练较慢，有很大的改进空间。

总体结果如下：

F1-score(三类平均)	A1	A3	B2	B3
Neural network	0.49	0.49	0.56	0.60
LSTM	0.45	0.48	0.59	0.57

可以看到，结果明显比线性分类器（0.40 左右）要好很多。

而两类方法的结果也较为相似。但是，LSTM 如果进一步改进的话，结果应该会更好。因为价格数据作为时序数据，很适合 LSTM 来处理。

但是，如果与价格预测 1.5 中的 kNN 相比（0.61，0.58，0.74，0.73）的话，两者目前仍有一定的差距。

而如果不使用默认参数，使用 3 中的最优参数（a=10，b=20，theta=0.175），再加上 bootstrp 的话，Neural network 的结果为：A2=（0.68，0.68，0.68）；B3=（0.76，0.76，0.75）（准确率、召回率、f1-score）。

如果再尝试 kNN 的话，则能够达到（0.87，0.87，0.87），仍然更优一些。

（当然此时的测试集本身也很小了，每类只有几百个）。

Neural Network 的的详细结果如下：（默认参数）

在训练集上：

Neural Network	A1	A3	B2	B3
混淆矩阵（每行为实际的类，每列为预测的类）	**16298** 2636 3731 3833 **15438** 3394 5267 2862 **14536**	**19046** 4134 4053 4071 **19073** 4089 6852 3708 **16673**	**2801** 1290 1223 574 **3838** 902 730 1183 **3401**	**5847** 1027 810 1179 **5578** 927 1552 1524 **4608**
精确率	跌：.64 平：.74 涨：.67 均：.68	跌：.64 平：.71 涨：.67 均：.67	跌：.68 平：.61 涨：.62 均：.64	跌：.68 平：.69 涨：.73 均：.70
召回率	跌：.72 平：.68 涨：.64 均：.68	跌：.70 平：.70 涨：.61 均：.67	跌：.53 平：.72 涨：.64 均：.63	跌：.76 平：.73 涨：.60 均：.70
F1-score	跌：.68 平：.71 涨：.66 **均：.68**	跌：.67 平：.70 涨：.64 **均：.67**	跌：.59 平：.66 涨：.63 **均：.63**	跌：.72 平：.71 涨：.66 **均：.69**

在测试集上：

Neural Network	A1	A3	B2	B3
混淆矩阵（每行为实际的类，每列为预测的类）	**8034** 3375 3701 4091 **7173** 3846 4643 3635 **6832**	**9727** 4624 3805 4051 **9337** 4768 5584 4650 **7922**	**1637** 1059 847 494 **2264** 785 538 894 **2111**	**3365** 1039 720 1075 **3179** 870 1221 1251 **2652**
精确率	跌：.48 平：.51 涨：.48 均：.49	跌：.50 平：.50 涨：.48 均：.49	跌：.61 平：.54 涨：.56 均：.57	跌：.59 平：.58 涨：.63 均：.60
召回率	跌：.53 平：.47 涨：.45 均：.49	跌：.54 平：.51 涨：.44 均：.50	跌：.46 平：.64 涨：.60 均：.57	跌：.66 平：.62 涨：.52 均：.60
F1-score	跌：.50 平：.49 涨：.46 **均：.49**	跌：.52 平：.51 涨：.46 **均：.49**	跌：.53 平：.58 涨：.58 **均：.56**	跌：.62 平：.60 涨：.57 **均：.60**

可以看到，在训练集上的结果比测试集上会高；
另外，后两个类的预测结果好于前两个类，其中 B3 的结果最好。

3. 不同参数对结果的影响：

3.1 a,b 对结果的影响：

即用多长时间区间内的未来数据来对涨跌进行标注。

初始设为 a=10，b=40，既未来 5~20 秒的数据。

不同的 a，b 对结果的影响如下：（B3 合约，f1-score）：

	a=0	a=10	a=20
b=20	0.57	**0.67**	
b=40	0.59	0.60.	0.57
b=60	0.54	0.55	0.55

惊奇的发现，当选择 5~10s 的数据时，结果会显著改进，我想应该是因为期货数据变化迅速，过去对未来的影响只能维持很短时间。

3.2 theta 对结果的影响：

即判断涨跌的阈值。

初始为 0.15%

不同的 theta 对结果的影响如下：（B3 合约，f1-score）：

theta=0.1%	theta=0.15%	theta=0.2%
0.54%	0.60	0.65

可以看到如果阈值提高，结果会更好，当然数据规模也会更小。

这应该是因为当涨和跌的幅度更大时，其特征也就更明显，使得预测也变得更加容易。

3.3 dim 对结果的影响：

即用多久的历史数据来对涨跌进行预测。

初始设为 100，既过去 50s 的数据。

不同的 dim 对结果的影响如下：（B3 合约，f1-score）：

	dim=20	dim=60	dim=100	dim=180	dim=360
f1-score	0.47	0.53	0.60	0.63	0.66

可以看到，使用更长历史数据时的表现会更好，因为包含了更多的信息。但训练速度也会更慢。

3.4 多层神经网络结构对结果的影响：

分别对隐藏层为 1、2、3、4 时进行比较

	1	2	3	4
	(256,)	(256,128)	(256,128,64)	(256,128,64,32)

	0.49	0.56	0.60	0.60

可以看到层数越多效果越好，但是 3 到 4 的改进并不明显。因此 3 层就足够了。同时，增加每层的节点数对结果改进也不明显。

3.5 多层神经网络参数对结果的影响：

对结果有较大影响的参数有 alpha，beta_1 和 max_iter。前两者如果都使用默认参数的话结果会较好，而 max_iter 更大会使结果稍好一些。具体数据略。

3.6 重采样对结果的影响：

如果使用 sklearn 中的 bootstrap 方法，则能够使得结果进一步提高。即从每一类中有放回的采样一部分，用这一部分数据来训练分类器，采样多次得到多个分类器，最后投票来得到最终分类结果。具体改进见 2 中黑体部分。

4. 进一步改进的方向：

4.1 特征提取的改进：

除了价格数据，交易量数据也许会有一定的使用价值。除此之外，

4.2 数据集划分的改进：

当前的划分仍旧较为简单，可以考虑在更加真实的情况下进行预测。

4.3 验证方法的改进：

当前使用分类的准确率、召回率等，而为了模拟真实的期货交易，可以考虑通过回测来得到期望收益率，来更直接的对各个方法、参数等进行比较。

4.4 模型的改进：

LSTM 被证明在股票预测中的效果很好，如果能够合理运用的话，在这里效果应该不比 kNN 逊色。