Spring Boot Note

this repository is some notes about Spring Boot MVC AutoConfiguration

一、输入：

train.txt/test.txt

要求：

- **出租车id**必须为5位，比如“14276”，我原本的出租车id为“2012000002”，改为“00002”即可
- **出发网格坐标**格式为”69.0--/-14.0“，作者没有描述清楚
- **出发方向**”abcdefghi“，全部规定为e
- **出发时间**本数据集可以满足
- **预计寻客**时间不清楚怎样算
- **网格变化**没有表述清楚，我设置随机值
- **目标网格id**需要进行分类，类别数不得超过2823

```
## 数据样例
|出租车id|出发网格坐标|出发方向|出发时间|预计寻客时间|网格变化|目标网格id|
|:--:|:--:|:--:|:--:|:--:|:--:|:--:|
|14276|69.0-14.0|d|9.6|1050s|-7.0x2.0y|1677|
|80862|15.0-29.0|a|17.4|6s|0.0x0.0y|2378|
|41136|47.0--44.0|e|0.6|943s|-7.0x-2.0y|2607|
|80345|41.0-38.0|h|1.0|1124s|12.0x-13.0y|253|
|66746|49.0--14.0|b|23.0|354s|0.0x9.0y|2790|
|12195|39.0-33.0|d|12.8|1571s|-2.0x2.0y|2731|
```

二、如何处理：

1、预处理：

输入：train.txt/test.txt

输出：in_array, target_array, test_in_array, test_target_array, n_status

- in_array：为每一个训练数据编号

```
|出租车id|出发网格坐标|出发方向|出发时间|预计寻客时间|网格变化|
|:--:|:--:|:--:|:--:|:--:|:--:|
|1|4|7|10|13|16|
|2|5|8|11|14|17|
|3|6|9|12|15|18|

```

- target_array：对训练数据目标网格id分类后产生的数组
- test_in_array：为每一个测试数据编号

```
|出租车id|出发网格坐标|出发方向|出发时间|预计寻客时间|网格变化|
|:--:|:--:|:--:|:--:|:--:|:--:|
|1|4|7|10|13|16|
|2|5|8|11|14|17|
|3|6|9|12|15|18|
```

- test_target_array：对测试数据目标网格id分类后产生的数组
- n_status： 编号数量

2、嵌入并预测

- 训练过程：

```
2570/2570 [==============================] - 1s 518us/step - loss: 7.9458 - acc: 3.8911e-04 - val_loss: 7.9453 - val_acc: 0.0000e+00
Epoch 2/50
2570/2570 [==============================] - 1s 407us/step - loss: 7.9423 - acc: 0.0023 - val_loss: 7.9450 - val_acc: 0.0000e+00
Epoch 3/50
2570/2570 [==============================] - 1s 404us/step - loss: 7.9386 - acc: 0.0070 - val_loss: 7.9446 - val_acc: 0.0000e+00
Epoch 4/50
2570/2570 [==============================] - 1s 407us/step - loss: 7.9347 - acc: 0.0237 - val_loss: 7.9442 - val_acc: 0.0000e+00
Epoch 5/50
2570/2570 [==============================] - 1s 401us/step - loss: 7.9296 - acc: 0.0595 - val_loss: 7.9437 - val_acc: 0.0000e+00
Epoch 6/50
2570/2570 [==============================] - 1s 398us/step - loss: 7.9238 - acc: 0.0977 - val_loss: 7.9430 - val_acc: 0.0000e+00
Epoch 7/50
2570/2570 [==============================] - 1s 395us/step - loss: 7.9168 - acc: 0.1533 - val_loss: 7.9423 - val_acc: 0.0000e+00
Epoch 8/50
2570/2570 [==============================] - 1s 393us/step - loss: 7.9088 - acc: 0.2175 - val_loss: 7.9413 - val_acc: 0.0000e+00
Epoch 9/50
2570/2570 [==============================] - 1s 394us/step - loss: 7.8996 - acc: 0.2580 - val_loss: 7.9402 - val_acc: 0.0000e+00
Epoch 10/50
2570/2570 [==============================] - 1s 392us/step - loss: 7.8893 - acc: 0.2786 - val_loss: 7.9389 - val_acc: 0.0000e+00
Epoch 11/50
2570/2570 [==============================] - 1s 395us/step - loss: 7.8772 - acc: 0.2798 - val_loss: 7.9373 - val_acc: 0.0000e+00
Epoch 12/50
2570/2570 [==============================] - 1s 392us/step - loss: 7.8640 - acc: 0.2981 - val_loss: 7.9355 - val_acc: 0.0000e+00
Epoch 13/50
2570/2570 [==============================] - 1s 394us/step - loss: 7.8483 - acc: 0.3148 - val_loss: 7.9336 - val_acc: 0.0000e+00
Epoch 14/50
2570/2570 [==============================] - 1s 398us/step - loss: 7.8321 - acc: 0.3113 - val_loss: 7.9314 - val_acc: 0.0000e+00
Epoch 15/50
```

```
2570/2570 [==============================] - 1s 400us/step - loss: 7.8132 - acc:
0.3257 - val_loss: 7.9290 - val_acc: 0.0000e+00
Epoch 16/50
2570/2570 [==============================] - 1s 397us/step - loss: 7.7930 - acc:
0.3331 - val_loss: 7.9264 - val_acc: 0.0000e+00
Epoch 17/50
2570/2570 [==============================] - 1s 393us/step - loss: 7.7695 - acc:
0.3518 - val_loss: 7.9236 - val_acc: 0.0000e+00
Epoch 18/50
2570/2570 [==============================] - 1s 392us/step - loss: 7.7451 - acc:
0.3533 - val_loss: 7.9207 - val_acc: 0.0000e+00
Epoch 19/50
2570/2570 [==============================] - 1s 392us/step - loss: 7.7186 - acc:
0.3533 - val_loss: 7.9176 - val_acc: 0.0000e+00
Epoch 20/50
2570/2570 [==============================] - 1s 395us/step - loss: 7.6892 - acc:
0.3658 - val_loss: 7.9143 - val_acc: 0.0000e+00
Epoch 21/50
2570/2570 [==============================] - 1s 399us/step - loss: 7.6578 - acc:
0.3599 - val_loss: 7.9112 - val_acc: 0.0000e+00
Epoch 22/50
2570/2570 [==============================] - 1s 396us/step - loss: 7.6253 - acc:
0.3556 - val_loss: 7.9083 - val_acc: 0.0000e+00
Epoch 23/50
2570/2570 [==============================] - 1s 405us/step - loss: 7.5891 - acc:
0.3669 - val_loss: 7.9056 - val_acc: 0.0000e+00
Epoch 24/50
2570/2570 [==============================] - 1s 404us/step - loss: 7.5505 - acc:
0.3864 - val_loss: 7.9034 - val_acc: 0.0010
Epoch 25/50
2570/2570 [==============================] - 1s 407us/step - loss: 7.5099 - acc:
0.3875 - val_loss: 7.9014 - val_acc: 0.0010
Epoch 26/50
2570/2570 [==============================] - 1s 411us/step - loss: 7.4679 - acc:
0.3930 - val_loss: 7.8996 - val_acc: 0.0010
Epoch 27/50
2570/2570 [==============================] - 1s 569us/step - loss: 7.4213 - acc:
0.4148 - val_loss: 7.8981 - val_acc: 0.0010
Epoch 28/50
2570/2570 [==============================] - 1s 406us/step - loss: 7.3736 - acc:
0.4245 - val_loss: 7.8966 - val_acc: 0.0010
Epoch 29/50
2570/2570 [==============================] - 1s 406us/step - loss: 7.3210 - acc:
0.4385 - val_loss: 7.8952 - val_acc: 0.0010
Epoch 30/50
2570/2570 [==============================] - 1s 399us/step - loss: 7.2671 - acc:
0.4642 - val_loss: 7.8938 - val_acc: 0.0010
Epoch 31/50
2570/2570 [==============================] - 1s 398us/step - loss: 7.2095 - acc:
0.4981 - val_loss: 7.8928 - val_acc: 0.0000e+00
Epoch 32/50
2570/2570 [==============================] - 1s 399us/step - loss: 7.1463 - acc:
0.5304 - val_loss: 7.8920 - val_acc: 0.0010
Epoch 33/50
2570/2570 [==============================] - 1s 393us/step - loss: 7.0790 - acc:
0.5689 - val_loss: 7.8912 - val_acc: 0.0010
Epoch 34/50
```

```
2570/2570 [==============================] - 1s 400us/step - loss: 7.0095 - acc:
0.6008 - val_loss: 7.8904 - val_acc: 0.0010
Epoch 35/50
2570/2570 [==============================] - 1s 407us/step - loss: 6.9372 - acc:
0.6533 - val_loss: 7.8898 - val_acc: 0.0010
Epoch 36/50
2570/2570 [==============================] - 1s 399us/step - loss: 6.8581 - acc:
0.6938 - val_loss: 7.8895 - val_acc: 0.0010
Epoch 37/50
2570/2570 [==============================] - 1s 394us/step - loss: 6.7736 - acc:
0.7241 - val_loss: 7.8892 - val_acc: 0.0010
Epoch 38/50
2570/2570 [==============================] - 1s 395us/step - loss: 6.6853 - acc:
0.7720 - val_loss: 7.8892 - val_acc: 0.0000e+00
Epoch 39/50
2570/2570 [==============================] - 1s 396us/step - loss: 6.5894 - acc:
0.8132 - val_loss: 7.8893 - val_acc: 0.0010
Epoch 40/50
2570/2570 [==============================] - 1s 396us/step - loss: 6.4874 - acc:
0.8502 - val_loss: 7.8898 - val_acc: 0.0000e+00
Epoch 41/50
2570/2570 [==============================] - 1s 397us/step - loss: 6.3876 - acc:
0.8786 - val_loss: 7.8904 - val_acc: 0.0000e+00
Epoch 42/50
2570/2570 [==============================] - 1s 395us/step - loss: 6.2758 - acc:
0.9019 - val_loss: 7.8913 - val_acc: 0.0000e+00
Epoch 43/50
2570/2570 [==============================] - 1s 394us/step - loss: 6.1626 - acc:
0.9152 - val_loss: 7.8928 - val_acc: 0.0000e+00
Epoch 44/50
2570/2570 [==============================] - 1s 392us/step - loss: 6.0385 - acc:
0.9350 - val_loss: 7.8948 - val_acc: 0.0000e+00
Epoch 45/50
2570/2570 [==============================] - 1s 394us/step - loss: 5.9031 - acc:
0.9537 - val_loss: 7.8972 - val_acc: 0.0010
Epoch 46/50
2570/2570 [==============================] - 1s 392us/step - loss: 5.7697 - acc:
0.9615 - val_loss: 7.9002 - val_acc: 0.0010
Epoch 47/50
2570/2570 [==============================] - 1s 394us/step - loss: 5.6253 - acc:
0.9626 - val_loss: 7.9038 - val_acc: 0.0010
Epoch 48/50
2570/2570 [==============================] - 1s 397us/step - loss: 5.4822 - acc:
0.9700 - val_loss: 7.9078 - val_acc: 0.0010
Epoch 49/50
2570/2570 [==============================] - 1s 397us/step - loss: 5.3287 - acc:
0.9747 - val_loss: 7.9122 - val_acc: 0.0010
Epoch 50/50
2570/2570 [==============================] - 1s 399us/step - loss: 5.1745 - acc:
0.9817 - val_loss: 7.9167 - val_acc: 0.0000e+00
```

- 预测结果：

  ```
  prob = model.predict(test_in_array, batch_size=batch_size)
  ```

```
[[  3.64165717e-05   2.74343794e-04   4.14482405e-04 ...,   3.78384830e-05
    4.19457101e-05   4.13001580e-05]
 [  3.27585985e-05   2.59199296e-04   3.99807584e-04 ...,   3.12815464e-05
    3.27650960e-05   3.39298749e-05]
 [  4.90684688e-05   3.99563200e-04   2.82602821e-04 ...,   4.46101440e-05
    4.80465133e-05   4.82552787e-05]
 ...,
 [  3.68002620e-05   2.34481893e-04   5.22579183e-04 ...,   3.41883351e-05
    3.53059768e-05   3.54943259e-05]
 [  4.16924813e-05   2.40768190e-04   4.65696328e-04 ...,   4.32559173e-05
    3.96053256e-05   4.09918248e-05]
 [  4.50075604e-05   2.86251132e-04   1.95218134e-04 ...,   4.85109158e-05
    3.53059768e-05   3.54943259e-05]
```

- 从预测结果中选100个出现次数最多的值：

  ```
  target_classes = [np.argmax(class_list) for class_list in prob]
  print(Counter(target_classes).most_common(100))
  ```

二、输出：

输出两个结果：

1、一个hdf5文件：

作用：用来储存每个周期该模型的信息

存储操作：

```
model = embedding_mlp(n_status)
checkpoint = ModelCheckpoint('models/weigts.{epoch:02d}-{val_acc:.2f}.hdf5',
monitor='val_acc', verbose=0,save_best_only=True, mode='auto')
```

ModelCheckpoint的源码：

```
class ModelCheckpoint(Callback):
    """Save the model after every epoch.

    `filepath` can contain named formatting options,
    which will be filled the value of `epoch` and
    keys in `logs` (passed in `on_epoch_end`).

    For example: if `filepath` is `weights.{epoch:02d}-{val_loss:.2f}.hdf5`,
    then the model checkpoints will be saved with the epoch number and
    the validation loss in the filename.

    # Arguments
        filepath: string, path to save the model file.
        monitor: quantity to monitor.
        verbose: verbosity mode, 0 or 1.
        save_best_only: if `save_best_only=True`,
            the latest best model according to
            the quantity monitored will not be overwritten.
        mode: one of {auto, min, max}.
            If `save_best_only=True`, the decision
```

```
            to overwrite the current save file is made
            based on either the maximization or the
            minimization of the monitored quantity. For `val_acc`,
            this should be `max`, for `val_loss` this should
            be `min`, etc. In `auto` mode, the direction is
            automatically inferred from the name of the monitored quantity.
        save_weights_only: if True, then only the model's weights will be
            saved (`model.save_weights(filepath)`), else the full model
            is saved (`model.save(filepath)`).
        period: Interval (number of epochs) between checkpoints.
    """

    def __init__(self, filepath, monitor='val_loss', verbose=0,
                 save_best_only=False, save_weights_only=False,
                 mode='auto', period=1):
        super(ModelCheckpoint, self).__init__()
        self.monitor = monitor
        self.verbose = verbose
        self.filepath = filepath
        self.save_best_only = save_best_only
        self.save_weights_only = save_weights_only
        self.period = period
        self.epochs_since_last_save = 0

        if mode not in ['auto', 'min', 'max']:
            warnings.warn('ModelCheckpoint mode %s is unknown, '
                          'fallback to auto mode.' % (mode),
                          RuntimeWarning)
            mode = 'auto'

        if mode == 'min':
            self.monitor_op = np.less
            self.best = np.Inf
        elif mode == 'max':
            self.monitor_op = np.greater
            self.best = -np.Inf
        else:
            if 'acc' in self.monitor or self.monitor.startswith('fmeasure'):
                self.monitor_op = np.greater
                self.best = -np.Inf
            else:
                self.monitor_op = np.less
                self.best = np.Inf

    def on_epoch_end(self, epoch, logs=None):
        logs = logs or {}
        self.epochs_since_last_save += 1
        if self.epochs_since_last_save >= self.period:
            self.epochs_since_last_save = 0
            filepath = self.filepath.format(epoch=epoch + 1, **logs)
            if self.save_best_only:
                current = logs.get(self.monitor)
                if current is None:
                    warnings.warn('Can save best model only with %s available, '
                                  'skipping.' % (self.monitor), RuntimeWarning)
                else:
                    if self.monitor_op(current, self.best):
                        if self.verbose > 0:
                            print('\nEpoch %05d: %s improved from %0.5f to
%0.5f,'
                                  ' saving model to %s'
                                  % (epoch + 1, self.monitor, self.best,
                                     current, filepath))
                        self.best = current
                        if self.save_weights_only:
                            self.model.save_weights(filepath, overwrite=True)
                        else:
                            self.model.save(filepath, overwrite=True)
                    else:
                        if self.verbose > 0:
                            print('\nEpoch %05d: %s did not improve' %
                                  (epoch + 1, self.monitor))
            else:
                if self.verbose > 0:
                    print('\nEpoch %05d: saving model to %s' % (epoch + 1,
filepath))
                if self.save_weights_only:
                    self.model.save_weights(filepath, overwrite=True)
                else:
                    self.model.save(filepath, overwrite=True)
```

2、如上所示，k个元素的二元组

（目的地类，出现次数）

```
[(2298, 14), (1085, 8), (1200, 8), (2260, 8), (1751, 8), (2055, 7), (920, 7),
(484, 7), (831, 7), (887, 7), (2149, 6), (2297, 6), (2338, 6), (2426, 6), (600,
6), (2158, 6), (426, 5), (438, 5), (663, 5), (1800, 5), (167, 5), (934, 5),
 (2050, 4), (2566, 4), (1125, 4), (1130, 4), (2159, 4), (1168, 4), (2303, 4),
(286, 4), (299, 4), (2434, 4), (1490, 4), (255, 4), (544, 4), (2139, 4), (548,
4), (756, 4), (1799, 4), (1842, 4), (1852, 4), (868, 4), (2048, 3), (1033, 3
), (2077, 3), (68, 3), (2118, 3), (83, 3), (86, 3), (1171, 3), (2196, 3), (1223,
3), (1197, 3), (2223, 3), (1403, 3), (1262, 3), (2291, 3), (2319, 3), (273, 3),
(292, 3), (294, 3), (295, 3), (363, 3), (2443, 3), (2444, 3), (1496, 3),
 (2524, 3), (1503, 3), (480, 3), (496, 3), (2554, 3), (516, 3), (358, 3), (526,
3), (1554, 3), (564, 3), (580, 3), (629, 3), (1668, 3), (1732, 3), (2340, 3),
(1762, 3), (740, 3), (1781, 3), (1806, 3), (790, 3), (1873, 3), (1899, 3),
(1948, 3), (1998, 3), (2020, 3), (2039, 3), (1016, 3), (2043, 3), (1028, 2), (13,
2), (1040, 2), (24, 2), (41, 2), (349, 2)]
```