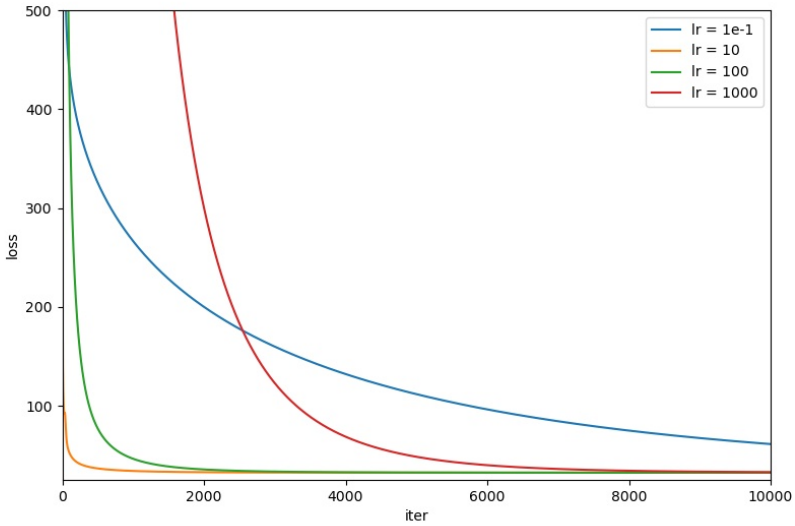
學號：**B06902125** 系級：資工三 姓名： 黃柏瑋

1. (2%)

使用四種不同的 **learning rate** 進行 **training (**其他參數需一致**)**，作圖並討論其收斂過程（橫軸為 **iteration** 次數，縱軸為 **loss** 的大小，四種 **learning rate** 的收斂線請以不同顏色呈現在一張圖裡做比較）。



- 由上圖可知，當lr =10時收斂速度最快。
- 若lr再大一些，會導致前面幾個iteration的步伐過大，loss暴增，反而得花更多時間才能收斂，當lr=1000時最為明顯。
- 若lr再小一些，雖然方向沒有偏掉但收斂的速度過慢，因此也需要花上更多時間才能收斂。

2. (1%)

比較取前 **5 hrs** 和前 **9 hrs** 的資料（**5*18 + 1 v.s 9*18 + 1**）在 **validation set** 上預測的結果，並說明造成的可能原因。

(5hrs)

```
iteration = 9987, loss = 34.75638374556188, val_loss = 30.932614338019963
iteration = 9988, loss = 34.75638195902599, val_loss = 30.932616685069576
iteration = 9989, loss = 34.75638017276717, val_loss = 30.932619031931612
iteration = 9990, loss = 34.75637838678541, val_loss = 30.932621378606086
iteration = 9991, loss = 34.75637660108064, val_loss = 30.932623725093062
iteration = 9992, loss = 34.75637481565281, val_loss = 30.932626071392587
iteration = 9993, loss = 34.756373030501884, val_loss = 30.932628417504677
iteration = 9994, loss = 34.756371245627825, val_loss = 30.932630763429398
iteration = 9995, loss = 34.75636946103056, val_loss = 30.932633109166776
iteration = 9996, loss = 34.756367676710056, val_loss = 30.932635454716852
iteration = 9997, loss = 34.756365892666274, val_loss = 30.932637800079696
iteration = 9998, loss = 34.75636410889916, val_loss = 30.932640145255316
iteration = 9999, loss = 34.75636232540868, val_loss = 30.932642490243772
```

(9hrs)

```
iteration = 9987, loss = 33.11478223735341, val_loss = 30.286505358657077
iteration = 9988, loss = 33.114774163791154, val_loss = 30.286517086924928
iteration = 9989, loss = 33.11476609328985, val_loss = 30.286528817378578
iteration = 9990, loss = 33.11475802584774, val_loss = 30.28654055001521
iteration = 9991, loss = 33.11474996146304, val_loss = 30.286552284832023
iteration = 9992, loss = 33.11474190013399, val_loss = 30.286564021826198
iteration = 9993, loss = 33.11473384185882, val_loss = 30.28657576099494
iteration = 9994, loss = 33.114725786635766, val_loss = 30.286587502335426
iteration = 9995, loss = 33.11471773446306, val_loss = 30.28659924584487
iteration = 9996, loss = 33.11470968533895, val_loss = 30.28661099152048
iteration = 9997, loss = 33.114701639261654, val_loss = 30.286622739359444
iteration = 9998, loss = 33.11469359622943, val_loss = 30.286634489358978
iteration = 9999, loss = 33.11468555624052, val_loss = 30.286646241516276
```

很顯然地，就validation loss來說，看了前9個小時的結果比看了前5個小時的還要好。這可能代表9小時前的資訊對於當前PM2.5也有一定的影響力，因此在我們給模型更多有用的資訊之後，模型能夠從中找到更詳細的估算依據，預測出更貼近真實值的答案。

3. (1%)

比較只取前 **9 hrs** 的 **PM2.5** 和取所有前 **9 hrs** 的 **features**（**9*1 + 1 vs. 9*18 + 1**）在 **validation set**上預測的結果，並說明造成的可能原因。

(PM2.5)

```
iteration = 9987, loss = 38.69040948061254, val_loss = 32.88473553385906
iteration = 9988, loss = 38.69040948061254, val_loss = 32.88473553385906
iteration = 9989, loss = 38.69040948061254, val_loss = 32.88473553385906
iteration = 9990, loss = 38.69040948061254, val_loss = 32.88473553385906
iteration = 9991, loss = 38.69040948061254, val_loss = 32.88473553385905
iteration = 9992, loss = 38.69040948061254, val_loss = 32.88473553385906
iteration = 9993, loss = 38.69040948061254, val_loss = 32.88473553385906
iteration = 9994, loss = 38.69040948061254, val_loss = 32.88473553385906
iteration = 9995, loss = 38.690409480612544, val_loss = 32.88473553385905
iteration = 9996, loss = 38.690409480612544, val_loss = 32.88473553385906
iteration = 9997, loss = 38.690409480612544, val_loss = 32.88473553385906
iteration = 9998, loss = 38.690409480612544, val_loss = 32.88473553385906
iteration = 9999, loss = 38.690409480612544, val_loss = 32.88473553385906
```

(all features)

```
iteration = 9987, loss = 33.11478223735341, val_loss = 30.286505358657077
iteration = 9988, loss = 33.114774163791154, val_loss = 30.286517086924928
iteration = 9989, loss = 33.11476609328985, val_loss = 30.286528817378578
iteration = 9990, loss = 33.11475802584774, val_loss = 30.28654055001521
iteration = 9991, loss = 33.11474996146304, val_loss = 30.286552284832023
iteration = 9992, loss = 33.11474190013399, val_loss = 30.286564021826198
iteration = 9993, loss = 33.11473384185882, val_loss = 30.28657576099494
iteration = 9994, loss = 33.114725786635766, val_loss = 30.286587502335426
iteration = 9995, loss = 33.11471773446306, val_loss = 30.28659924584487
iteration = 9996, loss = 33.11470968533895, val_loss = 30.28661099152048
iteration = 9997, loss = 33.114701639261654, val_loss = 30.286622739359444
iteration = 9998, loss = 33.11469359622943, val_loss = 30.286634489358978
iteration = 9999, loss = 33.11468555624052, val_loss = 30.286646241516276
```

由上圖可知，看了所有features的結果比只看PM2.5的還要好。這可能是因為其他的某些測項其實也和PM2.5有關，而且有用的比沒用的影響力更大，因此當我們給模型看了所有feature時，模型預測的結果固然比較好。

4. (2%)

請說明你超越 **baseline** 的 **model(最後選擇在Kaggle**上提交的**)** 是如何實作的（例如：怎麼進行 **feature selection,** 有沒有做 **pre-processing**、**learning rate** 的調整、**advanced gradient descent** 技術、不同的 **model** 等等）。

第一個model：Linear regression with adagrad

- 根據第一題的觀察，我選用的learning rate為10。

- 接著，我發現無論是training data或是testing data中都藏有負值，這在各個測項中都相當不合理，因此我利用內插法將負值移除。舉例來說，若某段時間的PM2.5值為2, -1, -1, 7, 6，則經過調整，會變成2, 3.25, 4.5, 5.75, 7, 6。經過這樣的調整，linear regression的public test error從5.49降至5.45。
- 原本我有根據validation data挑出最好的模型，但後來發現資料數量過小，用一筆validation data作為依據不太合適，因此在換了不同的validation data觀察後發現當lr=10時，iteration大概在10000附近可以有較低且穩定的validation loss。於是當我把所有的training data以lr=10、iteration=10000訓練完後，linear regression的public test error從5.45降至5.43
- 最後，我加上L1-regularization(alpha=10)，linear regression的public test error從5.43降至5.42

第二個model: SVR

- 延續第一個模型處理負值的方法，但不使用linear regression，而使用sklearn裡面的SVR(Epsilon-Support Vector Regression)模型。好處是它在算loss時會忽略掉預測值較準的data point，努力讓其他的data point更準一些，因此更能找出training data大方向上的趨勢。SVR的public test error大約為5.41，比linear regression的結果更好一些。