Linear kernel:

Before parameter optimizing :

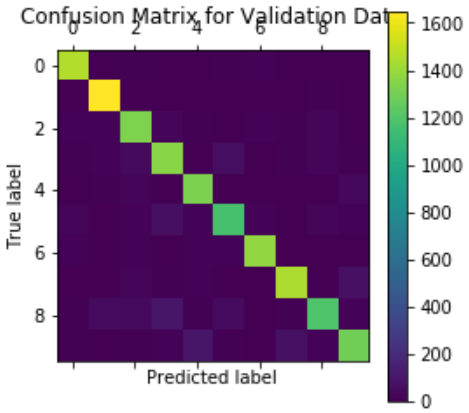
SVM Classifier with gamma = 0.1; Kernel = linear
Accuracies of the different cross validations
[0.9072592592592592, 0.9087407407407407, 0.9145185185185185, 0.9081345384501408]
average accuracy 0.9096632642421648
best parameter for linear kernel: C = 0.1
0.847 (+/-0.065) for {'C': 0.1, 'kernel': 'linear'}
0.847 (+/-0.065) for {'C': 0.2, 'kernel': 'linear'}
0.847 (+/-0.065) for {'C': 0.25, 'kernel': 'linear'}
0.847 (+/-0.065) for {'C': 0.5, 'kernel': 'linear'}
0.847 (+/-0.065) for {'C': 1, 'kernel': 'linear'}
0.847 (+/-0.065) for {'C': 100, 'kernel': 'linear'}

Changing the parameter C doesn't have an influence on the accuracy, so its better to choose a small C (only allow for few missclassifications). Because it doesn't appear to have an influence the accuracy with C = 1 and C = 0.1 are (approximately) the same. This might be the case because the linear kernel is overall not a good choice for the mnist data sat classification with SVM.

with optimized parameters: C=0.1

Accuracies of the different cross validations [0.9072592592592592, 0.9087407407407407, 0.9145185185185185, 0.9081345384501408]
average accuracy 0.9096632642421648
Accuracy of Classifier on Validation Images: 0.9080727951469902
Confusion Matrix:

```
[[1463    0    7    5    2   12   13    0    5    0]
 [   0 1651    3    5    1    3    2    3   10    1]
 [  13   17 1331   29    8    6   16   12   20    3]
 [  11   15   43 1360    2   59    1   10   32   11]
 [   6   10   20    1 1324    0    4    4    5   36]
 [  20    6   17   65   15 1173   18    2   30   13]
 [  14    3   17    0   14   18 1385    0    6    0]
 [   5    4   26    8   17    1    0 1449    3   77]
 [  10   46   43  100    7   47    6    8 1194   13]
 [   9    5    9   16   98    9    0   75   13 1292]]
```



RBF Kernel

Before Parameter tuning C = 1:

Accuracy [0.9370834444148227, 0.9437333333333333, 0.9464, 0.9445333333333333]
Average accuracy 0.9429375277703723

0.836 (+/-0.057) for {'C': 0.5, 'kernel': 'rbf'}
0.857 (+/-0.060) for {'C': 1, 'kernel': 'rbf'}
0.868 (+/-0.036) for {'C': 5, 'kernel': 'rbf'}
0.867 (+/-0.037) for {'C': 6, 'kernel': 'rbf'}
0.867 (+/-0.037) for {'C': 7, 'kernel': 'rbf'}
0.867 (+/-0.037) for {'C': 8, 'kernel': 'rbf'}
0.867 (+/-0.037) for {'C': 10, 'kernel': 'rbf'}

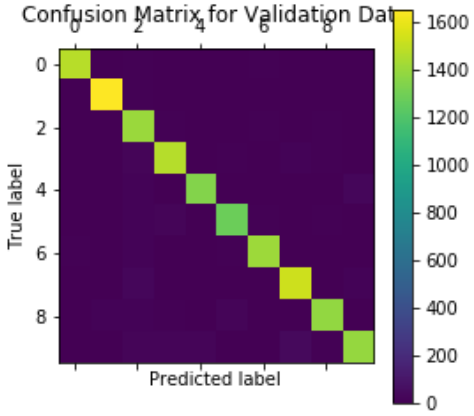
best parameters for rbf kernle C = 5

After Parameter tuning C = 5:
Accuracy [0.945881098373767, 0.9509333333333333, 0.9517333333333333, 0.9525333333333333]
Average accuracy 0.9502702745934417

Accuracy during cross validation with n= 4:
[0.9512592592592592, 0.9481481481481482, 0.9545185185185185, 0.9473996147577419]
Average accuracy in cross validation: 0.950331385170917

Accuracy of Classifier on Validation Images (test set): 0.9579361375908273
Confusion Matrix:

```
[[1476    0    9    1    1    2   11    1    4    2]
 [   0 1655   10    1    3    1    1    2    5    1]
 [   5    4 1408    8    5    0    9    6    8    2]
 [   1    5   23 1472    1   12    3   13    8    6]
 [   3    6   18    0 1349    2    4    0    2   26]
 [   6    3   13   22    5 1284   11    2    8    5]
 [  11    3   17    0    2    8 1412    0    4    0]
 [   2    2   30    1    6    1    0 1530    2   16]
 [   4   16   19    9    6   21    3    3 1388    5]
 [   6    5   22   20   21    5    0   40   11 1396]]
```



As can be seen from the accuracy the SVM with the rbf kernel is the better choice to classify the images

Source of large portions of the code: https://www.kaggle.com/soumya044/mnist-digit-recognizer-using-kernel-svm

Libraries: sklearn version 0.22.1