PreProcessing

### PCA

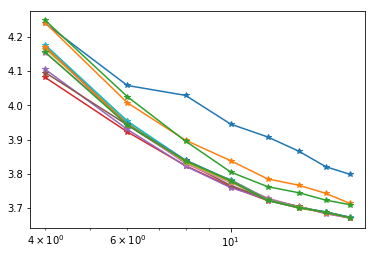
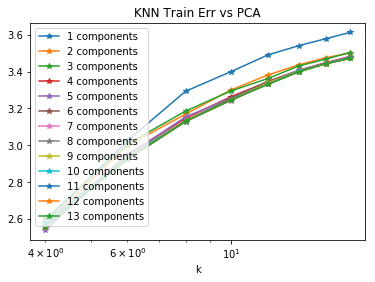
Feature 3 and feature 12 behaves relatively worse than other features after plotting all possible pairs of feature.

***Feature 3***

|  |  |  |  |
| --- | --- | --- | --- |
| C:\Users\Zhixuan\AppData\Local\Microsoft\Windows\INetCache\Content.Word\output_9_2.png | C:\Users\Zhixuan\AppData\Local\Microsoft\Windows\INetCache\Content.Word\output_9_14.png | C:\Users\Zhixuan\AppData\Local\Microsoft\Windows\INetCache\Content.Word\output_9_25.png | C:\Users\Zhixuan\AppData\Local\Microsoft\Windows\INetCache\Content.Word\output_9_44.png |

***Feature 12***

|  |  |  |  |
| --- | --- | --- | --- |
| C:\Users\Zhixuan\AppData\Local\Microsoft\Windows\INetCache\Content.Word\output_9_23.png | C:\Users\Zhixuan\AppData\Local\Microsoft\Windows\INetCache\Content.Word\output_9_34.png | C:\Users\Zhixuan\AppData\Local\Microsoft\Windows\INetCache\Content.Word\output_9_44.png | C:\Users\Zhixuan\AppData\Local\Microsoft\Windows\INetCache\Content.Word\output_9_53.png |
| C:\Users\Zhixuan\AppData\Local\Microsoft\Windows\INetCache\Content.Word\output_9_61.png | C:\Users\Zhixuan\AppData\Local\Microsoft\Windows\INetCache\Content.Word\output_9_68.png | C:\Users\Zhixuan\AppData\Local\Microsoft\Windows\INetCache\Content.Word\output_9_74.png | C:\Users\Zhixuan\AppData\Local\Microsoft\Windows\INetCache\Content.Word\output_9_79.png |
| C:\Users\Zhixuan\AppData\Local\Microsoft\Windows\INetCache\Content.Word\output_9_83.png | C:\Users\Zhixuan\AppData\Local\Microsoft\Windows\INetCache\Content.Word\output_9_86.png | C:\Users\Zhixuan\AppData\Local\Microsoft\Windows\INetCache\Content.Word\output_9_88.png | C:\Users\Zhixuan\AppData\Local\Microsoft\Windows\INetCache\Content.Word\output_9_90.png |



The left plot shows the Training Err vs the parameter k in K Nearest Neighbours. It shows that all features in the dataset are pretty useful. It is **NOT** really beneficial to ignore any feature while training a model. Therefore, all features are used while training a model.

### Outlier Exclusion

Below are plots of Ytr data vs. all 14 features respectively. Data points with rainfall in different ranges are plotted with different colors.

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |

After Excluding Outliers, Ytr data vs Features look like the following.

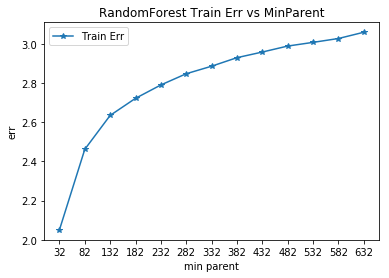
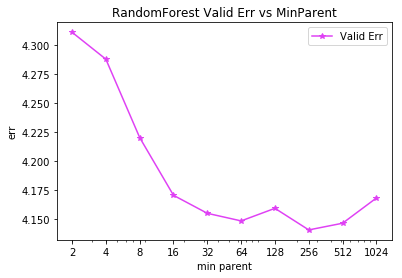
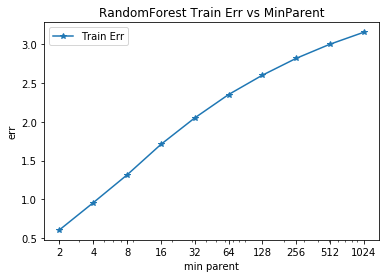
|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |

***\*\*Effectiveness:*** After applying outlier exclusion, the validation error of our KNN learner successfully dropped by more than 0.1.

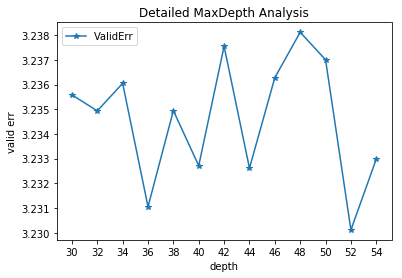
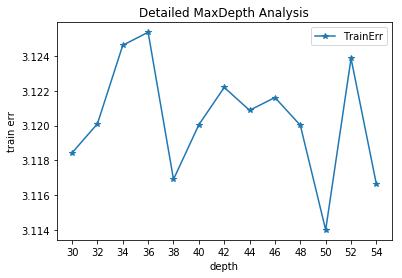
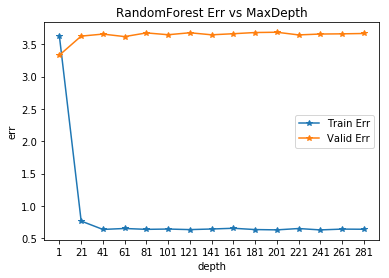
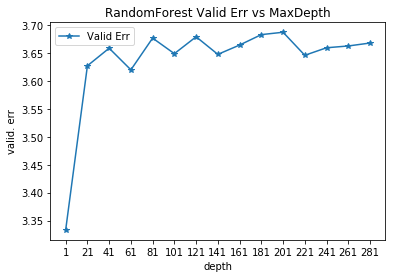
Random Forest

We first try to find the optimal parameters one-by-one using Cross-Validation.

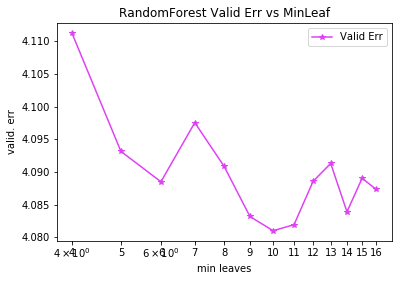
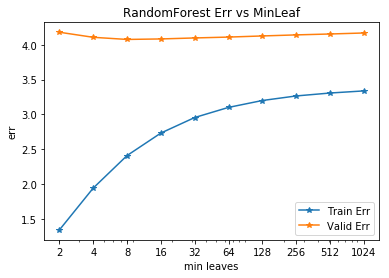
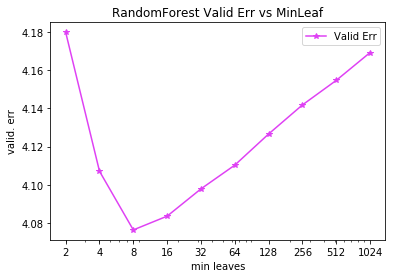
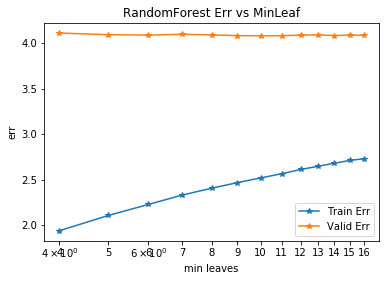
Finding best Min Parent



Finding Best MaxDepth. The correlation seems weak, so we used deep decision trees in our Random Forest Learner hoping for the ensemble process to take care of overfitting of individual decision trees in the random forest.



Finding the best MinLeaf value. The best minLeaf I found is 10.



Then we have two final random forest learners, one using the optimal parameters found above and the other is found by using the ***sklearn.model\_selection.randomizedSearchCV*** method to hunting for optimal combination of parameters using local random search.

The ***randomizedSearchCV*** methods generates a better random forest with lower validation error. Our guess is that the all parameters are entangled to each other, optimizing single parameters and simply combine them does not reach the global optimal.