محمد رضا جبلی
۹۴۳۱۰۳۵

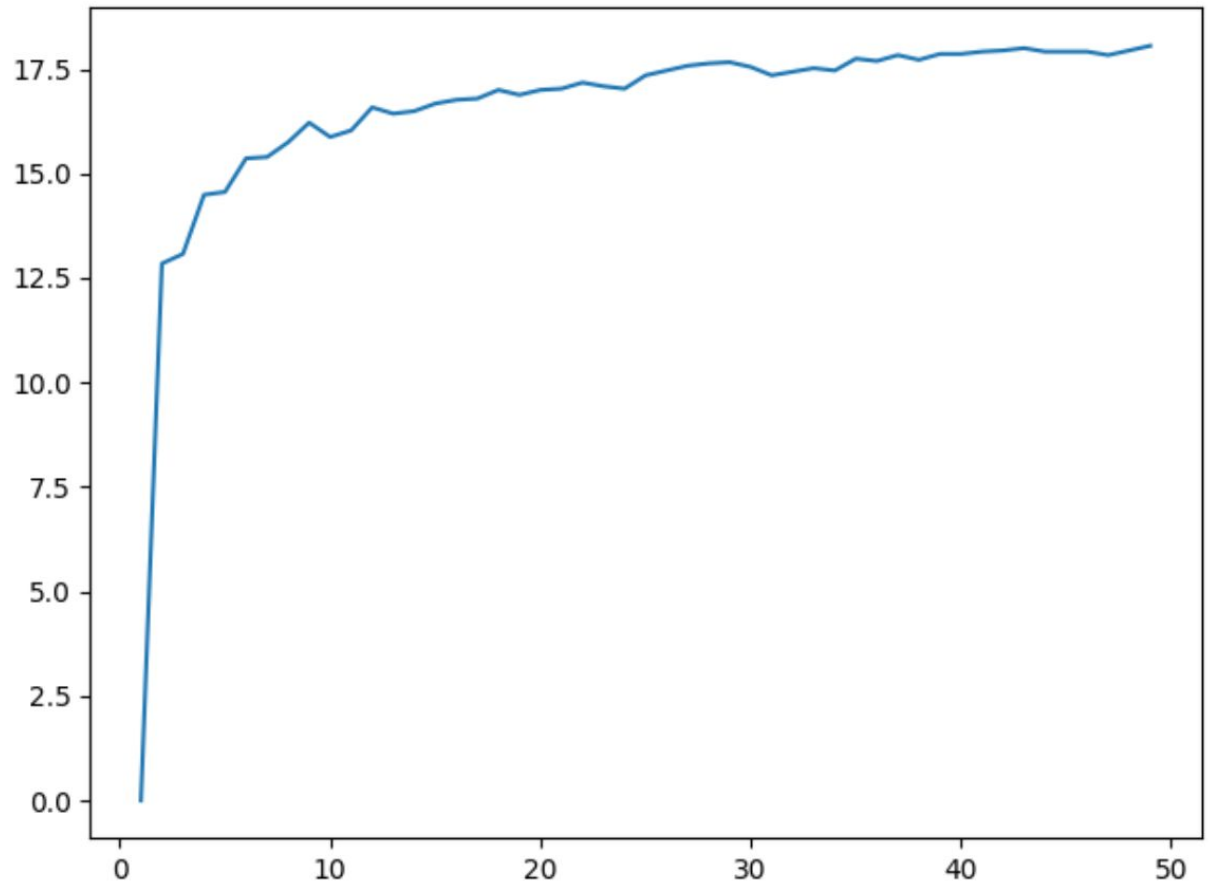
۷-

الف) چون از الگوریتم نزدیکترین همسایه استفاده میکنیم خطای دیتاست train منطقا ۰ خواهد شد چرا که هر داده به خودش نگاشت میشود.

```
/usr/local/bin/python /Users/rezajebeli97/PycharmProjects/DM_HW2_Q7/KNN.py
error :  0.0
```

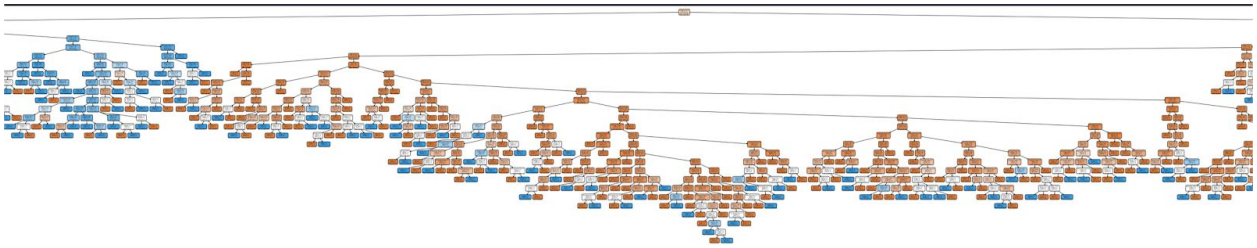
ب) از آنجا که داده ی train را predict میکنیم و خود classifier از داده های train ساخته شده پس هرچه از k بزرگتری در الگوریتم knn استفاده کنیم خطای بیشتری را متحمل خواهیم شد و بهترین حالت همان k=1 خواهد بود. اما اگر خطا روی داده های test را میخاستیم بررسی کنیم شرایط پیچیده تر بود و یک k میانی بدست می آمد که بهینه ترین حالت میشد.

نمودار زیر نشان دهنده ی میزان خطای پیشبینی داده های train میباشد. محور افقی نشان دهنده ی k و محور عمودی خطا است.



۸-

الف) شکل زیر قسمتی از درخت ساخته شده است. مشخص است که درخت ساده شده شاخه های بسیار زیادی دارد و به عبارتی overfit شده است.



دقت :

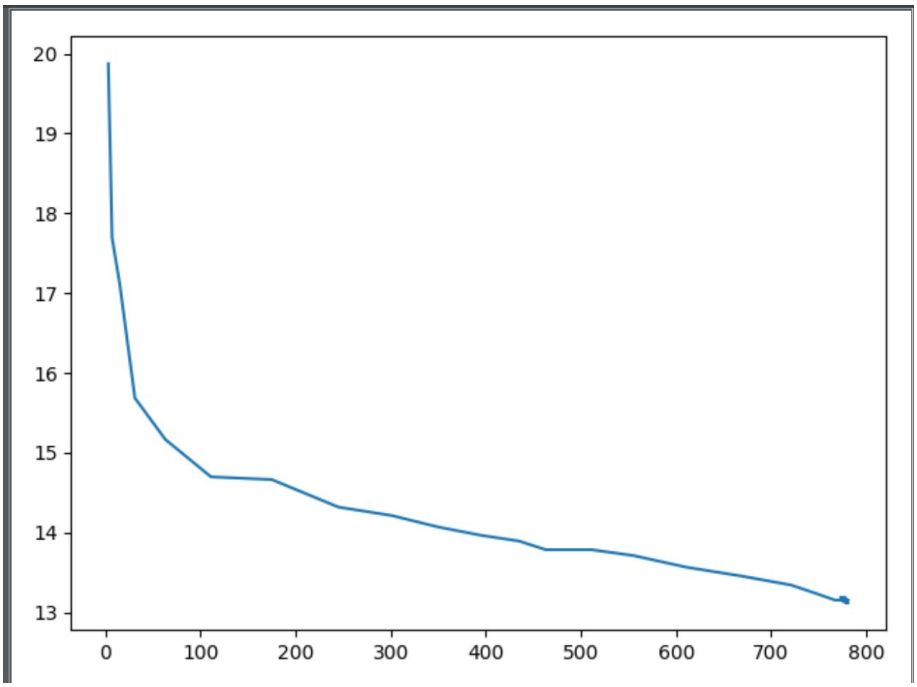
```
Euclidean error for train data :  0.0
Euclidean error for valid data :  18.33030277982336
Euclidean error for test data :  17.944358444926362
```

ب)

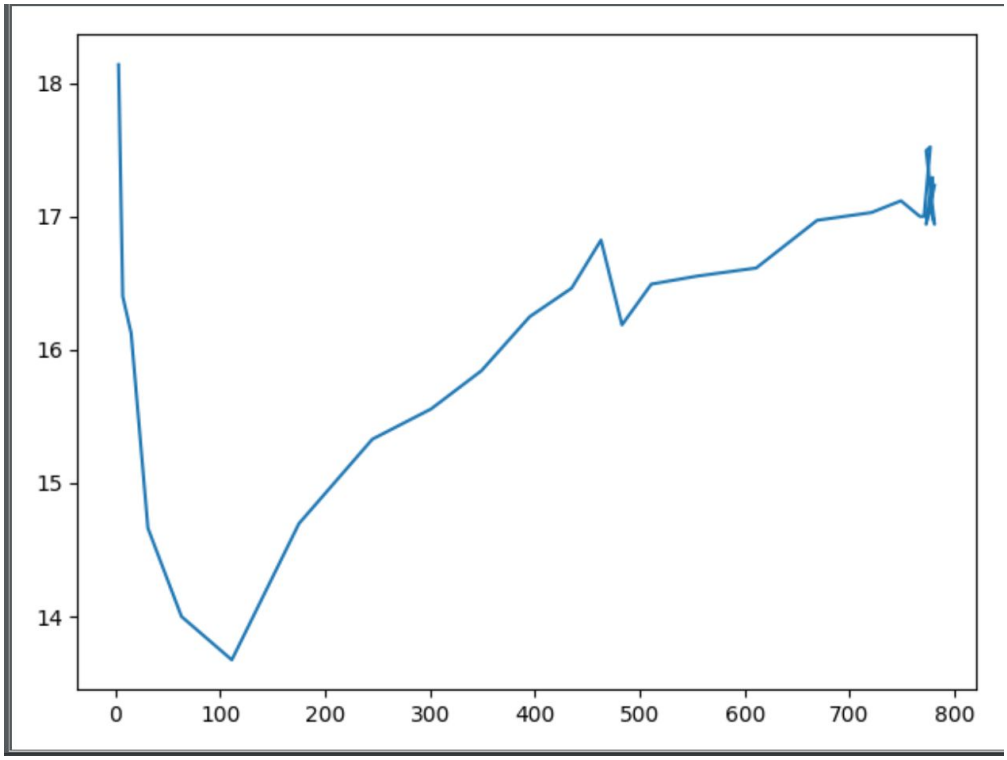
پیدا کردن تعداد نود مناسبی که خطای test , train , valid را کمینه کند از اهمیت بالایی برخوردار است

۳ شکل زیر در محور افقی تعداد نود و در محور عمودی خطای کلاسیفایر را نمایش میدهند که نمودار ها به ترتیب برای داده های test, valid, train می‌باشند

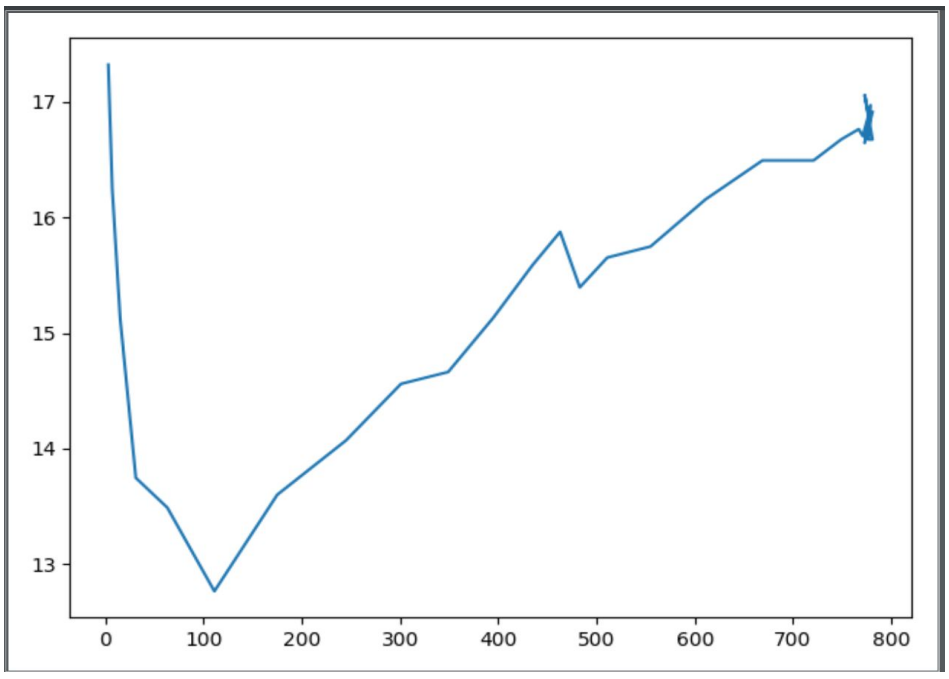
نمودار خطای داده های train برحسب تعداد نود که با افزایش آن کمتر شده است.



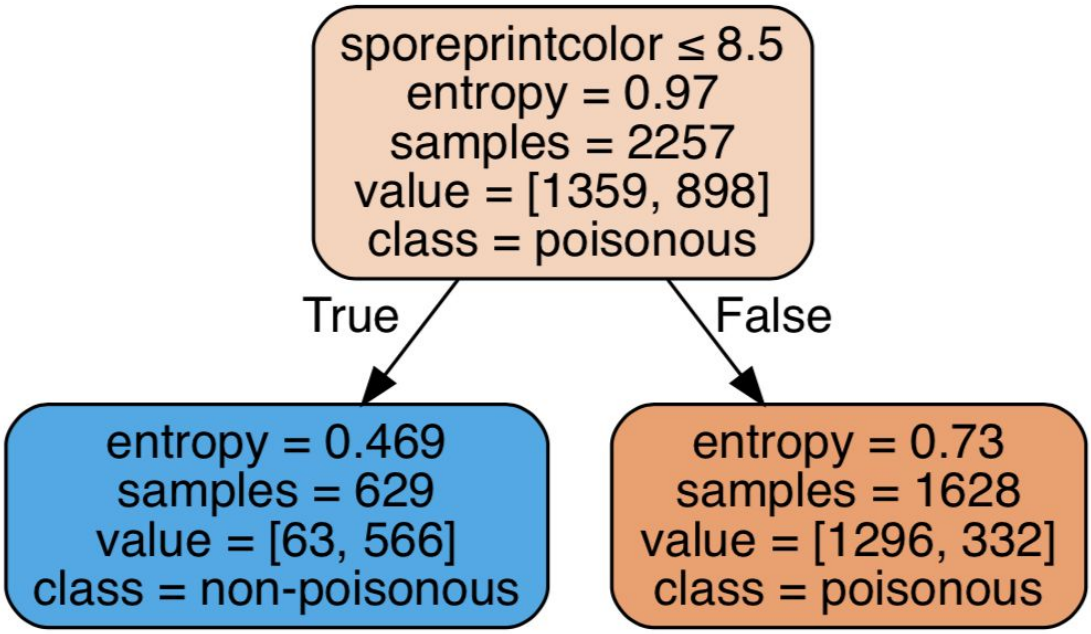
نمودار خطای داده های valid برحسب که حدودا با تعداد ۱۰۰ نود کمترین خطا را داشته و بعد از آن زیاد شده که نشان‌دهنده ی overfitting است.



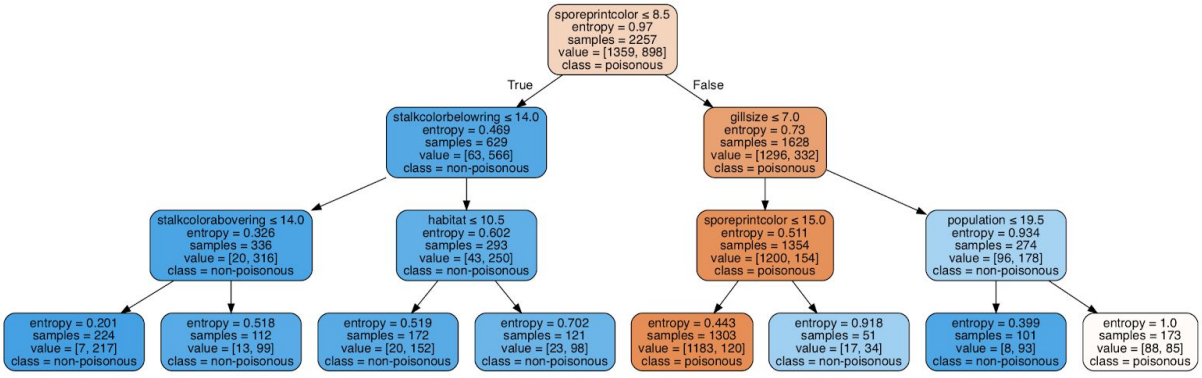
نمودار خطای داده های test برحسب که حدودا با تعداد ۱۰۰ نود کمترین خطا را داشته و بعد از آن زیاد شده که نشان‌دهنده ی overfitting است.



درخت ساخته شده با ۳ نود :



درخت ساخته شده با ۱۵ نود:



درخت ساخته شده با ۱۱۱ نود که حدودا بهینه ترین حالت میباشد.