Drug Activity Prediction

1. Name and ID:

SJSU ID: 011818729.
Rashmi Sharma

2. Rank and F1-score.

Rank : **5**
F1-score: 0.8485

3. **Approach**:

1. Data processing:
Loading the train.dat file and creating sparse matrix by denoting 1 in places where feature is present and 0 when feature is not present.
Found 1,00,000 attributes/features in train.dat file.
Separated labels from data.

2. Used dimensionality reduction method TruncatedSVD for reducing number of features from 1,00,000 to 1500 attributes.

3. Used SMOTE over sampler on reduced train data which samples using Synthetic Minority Over-sampling Technique i.e synthetically creating minority over samples using svm technique.

4. Used decision tree classifier with different weights given to 1 and 0 classes. 1 label is given more weightage (1.5) as this class is lesser in comparison to 0 class.

5. Used cross validation and F1-score to measure accuracy, as imbalanced data has best measure as F1.

6. Process test data using same truncated dimensionality reduction method as train data.

7. Performed classification on reduced test data.
F1-score on **training** data:

```
Report on Decision Tree
             precision    recall  f1-score   support

          0       0.95      0.92      0.94       722
          1       0.87      0.92      0.90       432

avg / total       0.92      0.92      0.92      1154

Cross validation scores: 0.92807765151
```

4. Methodology Of choosing Approach:

1. Since train data had 1,00,000 features, distribution was highly sparse. This was classic case of using dimensionality reduction. Since features were sparse, I used two methods SparsePCA and truncatedSVD which work best on high dimensionality data. I could see SparsePCA took lot of time and accuracy was quite low in comparison to truncatedSVD.

2. Data was imbalanced in nature, which meant minority class is not well represented and training on less data for that class may result in weak classifier. I searched on methods to tackle this issue and found we can use SMOTE over sampler, which performs over sampling by creating synthetic samples for minority class. I observed when I used kind='svm' accuracy increased significantly.

3. For classifiers I tried Random forest, Decision tree, SVM, Perceptron, AdaBoost, Naive bayes etc. I found highest accuracy achieved on Decision tree.
4. Hyper parameter random_state for classifier, I tried to increase from 42 to 91 in steps which added no value as such.

Classification report

No.	Method	Accuracy on test Data
1	LogisticRegression	0.41
2	AdaBoost	0.5926
3	**Decision Tree with weights**	**0.8485**
4	Decision Tree	0.8235
5	Random Forest	0.77
6	Perceptron	0.58
7	SVM	0.6897

References:

https://www.analyticsvidhya.com/blog/2017/03/imbalanced-classification-problem/
https://machinelearningmastery.com/tactics-to-combat-imbalanced-classes-in-your-machine-learning-dataset/