Lab 2 Report

Ben Fu
EE 385T Intro to Machine Learning

Code

https://github.com/bennycooly/INF_385T_Intro_To_ML/blob/master/labs/lab2

Questions

Note: I used the credit card fraud detection dataset found here:
https://archive.ics.uci.edu/ml/datasets/default+of+credit+card+clients

2. Hyperparameter Optimizations
    a. Decision Tree

Parameter	Values Tested	Optimal Value
criterion	gini, entropy	entropy
max_depth	1-10	5

    b. K-Nearest Neighbors

Parameter	Values Tested	Optimal Value
n_neighbors	1-100	35
metric	manhattan, euclidean	manhattan

    c. Support Vector Machine

Parameter	Values Tested	Optimal Value
C	0.001, 0.018, 0.316, 5.623, 10	0.001
degree	2, 3, 4	2
gamma	0.0001, 0.0003, 0.001, 0.003, 0.01	0.0001

Note: I used np.logspace() to distribute the values logarithmically instead of linearly

3. Model Evaluation
    c. Accuracy, precision, and recall

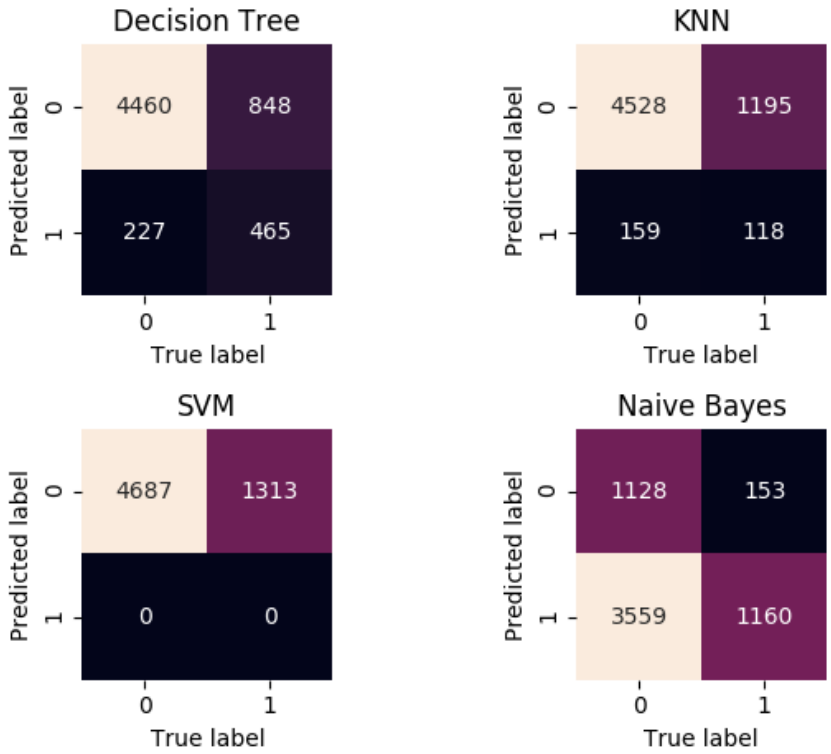
Model	Accuracy	Precision	Recall
Decision Tree	0.821	0.80	0.82

KNN	0.782	0.73	0.78
SVM	0.781	0.61	0.78
Naïve Bayes	0.381	0.74	0.38

Full output:

```
Decision Tree Metrics
Accuracy:  0.81
             precision    recall  f1-score   support

          0       0.83      0.96      0.89       311
          1       0.67      0.29      0.41        89

avg / total       0.79      0.81      0.78       400

K-Nearest Neighbors Metrics
Accuracy:  0.785
             precision    recall  f1-score   support

          0       0.79      0.99      0.88       311
          1       0.67      0.07      0.12        89

avg / total       0.76      0.79      0.71       400

SVM Metrics
Accuracy:  0.7775
C:\Program Files\Python36\lib\site-packages\sklearn\met
  'precision', 'predicted', average, warn_for)
             precision    recall  f1-score   support

          0       0.78      1.00      0.87       311
          1       0.00      0.00      0.00        89

avg / total       0.60      0.78      0.68       400

Naive Bayes Metrics
Accuracy:  0.37
             precision    recall  f1-score   support

          0       0.89      0.22      0.35       311
          1       0.25      0.91      0.39        89

avg / total       0.75      0.37      0.36       400
```

d. Confusion Matrices



e. Data Analysis

It looks like I definitely chose a hard dataset for classification. The dataset compares many different features such as age, sex, payments, and withdrawals and asks for a binary classification with 1 indicating that the person defaulted on their payment and 0 indicating that the person did not default on their payment. There were 30,000 total data points and 23 features, so this was definitely not an easy dataset to model. One thing I would like to mention is that the optimal values for the hyperparameters differed greatly when testing 1,000 to 10,000 data points. When testing over 10,000 data points, the SVM mode becomes incredibly slow (I stopped the tuning after 24 hours) because the runtime scales quadratically with the input points. Thus, the hyperparameters I tuned was for a partial dataset of 10,000 points. I then evaluated the models on the full dataset.

The decision tree seemed to consistently do better in every metric than the other models. A max depth of 5 definitely seems reasonable and looks like it is not overfitting the data. The model achieved more than 80% for accuracy, precision, and recall.

The KNN model was particularly interesting because the optimal value for k was the top value tested, 10 when I originally tested the values 1-10. I wanted to test this with a larger k so I tried setting the max value to 20, and surprisingly the model performed best at this new maximum value. After raising the limit to 100, the optimum value because 35. I think that a potential reason a larger value of k yields better accuracy is because of the large amount of data points being tested. I looked up a potential

starting value for k to be the square root of the number of data points, so it definitely makes sense that k is fairly large with this dataset.

The SVM model was interesting as well because it took the lowest values of C and gamma. Even more interesting is the fact that the model doesn't even try to predict the binary value 1. This is very surprising and I don't have a good explanation for why the accuracy doesn't seem to improve as the model actually makes predictions for 1. It could be the large number of features.

The Naïve Bayes model performed very poorly as expected. The model assumes independence between features, and in this case it makes a lot of sense that the features have correlations with other features.

Overall, I wish the dataset I chose was a bit simpler and more straightforward, but I was still able to obtain some really interesting results.