UDACITY

PROJECT

Finding Donors for CharityML

A part of the Machine Learning Engineer Nanodegree Program

PROJECT REVIEW

CODE REVIEW

NOTES

SHARE YOUR ACCOMPLISHMENT!

Requires Changes

7 SPECIFICATIONS REQUIRE CHANGES

Overall this is a good submission and shows that your project is on track. You're not that far off from completion, so keep at it! 😃

Exploring the Data

Student's implementation correctly calculates the following:

- **Number of records**
- **Number of individuals with income >$50,000**
- **Number of individuals with income <=$50,000**
- **Percentage of individuals with income > $50,000**

Great work getting the dataset stats!

Note: look at imbalanced target classes
As you can see we have an imbalanced proportion of individuals making more than $50k vs those making less, and will want to make sure the metric we're using for model evaluation is capturing how well the model is actually doing.

In this project we use the precision, recall, and F-beta scores, but we could also consider F1 score (which is equivalent to using F-beta with `beta=1`).

Preparing the Data

Student correctly implements one-hot encoding for the feature and income data.

Good job encoding the features and target labels!

We can also convert the `income` target labels to numerical values with `get_dummies` ...

```
pd.get_dummies(income_raw)['>50K']
```

Evaluating Model Performance

Student correctly calculates the benchmark score of the naive predictor for both accuracy and F1 scores.

Terrific work calculating the accuracy and F-score of a naive predictor — this will serve as a useful benchmark to evaluate how well our model is performing.

If we wanted to experiment with different values of `beta` for the F-beta score, we could also specify it with a variable and adjust as needed...

```
accuracy = greater_percent / 100.
recall = n_greater_50k / (n_greater_50k + 0.)
BETA = 0.5 # adjust the value as needed
fscore = (1 + BETA**2) * accuracy * recall / (BETA**2 * accuracy + recall)
```

The pros and cons or application for each model is provided with reasonable justification why each model was chosen to be explored.

Good discussion of the 3 models and why you chose them!

There are several issues to consider in choosing the best machine learning algorithm for your problem, and it's not always easy to know which model to use — with model selection it's often a good idea to try out simpler methods like Logistic Regression as a benchmark, and then move on to other approaches such as SVM, Decision Trees, and Ensemble methods.

Further reading:
You can also check out this guide from microsoft azure on choosing an algorithm, and this paper showing that rbf SVM and ensembles work best with binary classification.

Student successfully implements a pipeline in code that will train and predict on the supervised learning algorithm given.

Excellent work implementing the pipeline!

- With this pipeline you can see how the performance of the 3 models changes when using different training sizes passed to the `sample_size` parameter.
- You could also experiment on your own with making predictions on the training set using `sample_size`, but for the sake of speed we only use the first 300 training points here.

Student correctly implements three supervised learning models and produces a performance visualization.

Required: set a random state
Great work fitting the models, but make sure to follow the guidelines and also set random states for any classifiers where they can be specified.

Improving Results

Justification is provided for which model appears to be the best to use given computational cost, model performance, and the characteristics of the data.

Good justification of your choice of the SVM model by looking at factors such as the models' accuracy/F-scores and computational cost/time.

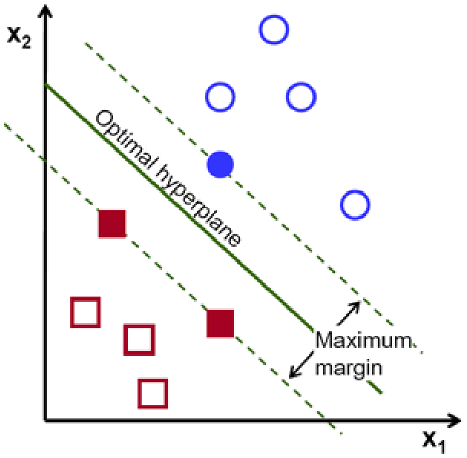
SVM is a good option to use here, and it's a good bet that we can improve the model's performance even further with some parameter tuning.

Student is able to clearly and concisely describe how the optimal model works in layman's terms to someone who is not familiar with machine learning nor has a technical background.

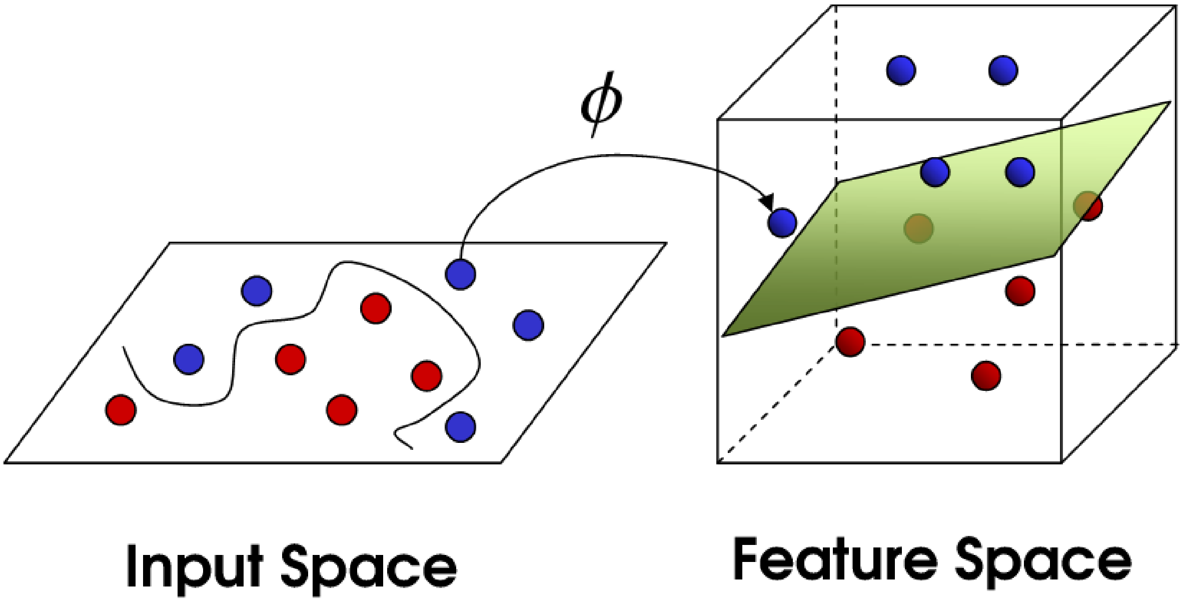
re: Question 4
Nice job explaining the SVM using an analogy, but make sure the discussion is comprehensive enough for a non-technical audience like CharityML to not only understand how the model is trained but also how it makes a prediction.

As an example, you could try to explain the model in the context of our current use case and mention that the SVM is trained with census information we have about individuals and their known incomes, and then describe how it will predict incomes for new individuals...

1. The SVM takes data about individuals whose census data is known (e.g., age, gender, etc) and uses them to create a function that draws a boundary between individuals with income over and under 50k. The boundary should be drawn so as to maximize the... *<<INSERT YOUR OWN DISCUSSION>>*



2. Often, though, it's not easy to draw a decision boundary in low dimensions, so the SVM separates the high & low income individuals by... *<<INSERT YOUR OWN DISCUSSION>>*



3. Using this function created with individuals we already know earn over or under 50k, the SVM can look at new potential donors' data and predict... *<<INSERT YOUR OWN DISCUSSION>>*

See below for more ideas on explaining SVM's:

- quora
- statsoft textbook
- http://blog.aylien.com/support-vector-machines-for-dummies-a-simple/

The final model chosen is correctly tuned using grid search with at least one parameter using at least three settings. If the model does not need any parameter tuning it is explicitly stated with reasonable justification.

n/a

Student reports the accuracy and F1 score of the optimized, unoptimized, and benchmark models correctly in the table provided. Student compares the final model results to previous results obtained.

n/a

Feature Importance

Student ranks five features which they believe to be the most relevant for predicting an individual's' income. Discussion is provided for why these features were chosen.

n/a

Student correctly implements a supervised learning model that makes use of the `feature_importances_` attribute. Additionally, student discusses the differences or similarities between the features they considered relevant and the reported relevant features.

n/a

Student analyzes the final model's performance when only the top 5 features are used and compares this performance to the optimized model from Question 5.

n/a

RESUBMIT

DOWNLOAD PROJECT

Learn the best practices for revising and resubmitting your project.

Have a question about your review? Email us at review-support@udacity.com and include the link to this review.

RETURN TO PATH

Rate this review

Student FAQ