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

8 SPECIFICATIONS REQUIRE CHANGES

Dear student,

Great submission 👍

I would recommend you to re-check the supervised models and understand how each model works and their tradeoffs.

Make sure you read the instructions before implementing the code.

Explanation of model in layman's terms demonstrates your understanding and easy for others to visualize how the model will do on given data set. Please check online resources and understand how your final model really works.

Take time and work each section and update the answers clearly. It is worth your time. I hope your next submission will meet project requirements.

keep up the good work! I look forward to next submission.

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**

you correctly calculated all the calculations. Good work!

Preparing the Data

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

Nice implementation using `get_dummies` method.

Evaluating Model Performance

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

The f-score is not calculated correctly. Please re-check and calculate it correctly.

You could check this link for further understanding precision and recall.

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

Please list all the references you use while listing out your pros and cons.

The reason of selection is not accurate for your selected models. We can apply any classification algorithm if the outcome is binary. But here the question is what makes you choose this particular model for the given data set?

Are you really sure all feature space is continuous?

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

> Fit the learner to the training data using slicing with 'sample_size'

You must fit your learner to training data using slicing with `sample_size`. You could write like :

```
learner = learner.fit(X_train[:sample_size], y_train[:sample_size])
```

> Compute F-score on the the first 300 training samples

results['f_train'] = fbeta_score(y_train[:300],predictions_train,beta=0.5)

> Compute F-score on the test set

```
results['f_test'] = fbeta_score(y_test,predictions_test,beta=0.5)
```

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

Nice implementation.

Suggestion

Please re-run the models after you correctly implemented the pipeline code.

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.

> Based on the three models selected, we recommend that the best model that is used for CharityML is Logistic Regression. This recommendation is based on the following.
> First, as compared the SVC and the Gradient Boosted Classifier, the Logistic Regression model is the quickest to train on 100% of the dataset. From the Model Training charts creating using the vs.evaluate(results, accuracy, fscore) function, there is an inverse relationship between the training times of the SVC and LogisticRegression. That means that as the dataset gets larger, the time to train the dataset get shorter for logistic regression and larger for the SVC.
> Second, there seems to be less overfitting with the logistic regression model as compared to the Gradient Boosting Classifier. Specifically, the difference in the F1 score, at 100% of the training data, is about 15 basis points for the Gradient Boosting Classifier, but about 5 points for the logistic regression model. In general, the degree to which a model overfits can be a cause of concern since overfitting is an indication of poor generalization (which is something that we strive to avoid when doing as a machine learning engineer).
> It is understood that the F1 score is higher for the Gradient Boosting Classifier, however, from the standpoint of overfitting, I have taken the position that is it is more important to have a model that

> does not overfit, even at the expense of performance. In an actual case, the decision to pick a model based on it's performance or over- or under-fitting is something that should be established before starting the project.

The description is not aligned with previous implemented models. You have not selected the Gradient boosting classifier but here you mentioned the same. Please re-check and update the description.

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.

Nice explanation.

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.

> Set a random_state if one is available to the same state you set before.

Make sure you use the same random_state as you previously defined for reproducing the results and your final model have better scores (atleast minor difference) compared to unoptimized model. You can test with different values for your tuning parameter to achive that.

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

Please update this section after you re-run your final model with random_state.

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.

Nice analysis and great selection of features.

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.

> If you were not close, why do you think these features are more relevant?

Please answer this question. Why do you think the result graph features are more relevant compared to your previous selected ones?

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.

> How does the final model's F-score and accuracy score on the reduced data using only five features compare to those same scores when all features are used?

You should answer this question.

I would recommend you to re-run this section acutally we are using the same final model here on both full and reduced data and I guess the results will change after you run your final model with random_state.

Please update the description as per results.

RESUBMIT

DOWNLOAD PROJECT

Best practices for your project resubmission

Ben shares 5 helpful tips to get you through revising and resubmitting your project.

Watch Video (3:01)

RETURN TO PATH

Rate this review

Student FAQ