‹ Back to Machine Learning Engineer Nanodegree

Finding Donors for CharityML

REVIEW | CODE REVIEW | HISTORY

Requires Changes

3 SPECIFICATIONS REQUIRE CHANGES

Dear student,

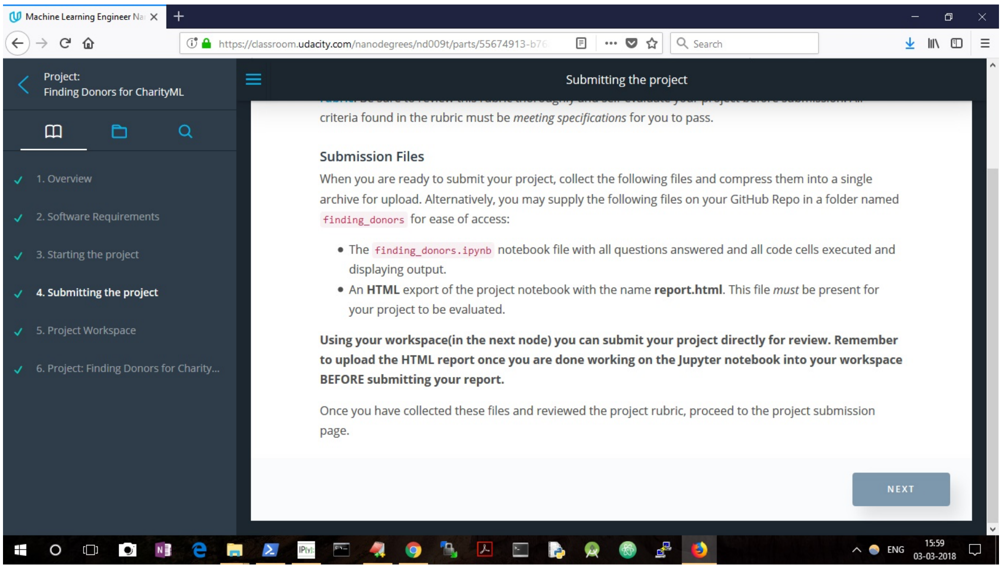
Really great submission 👍

I am really impressed with your analysis of each of model pros, cons and reason of selection but you missed few minor points and small code changes. I am sure they won't take much time. Please check each section and update as per suggestions, if provided.

Submission suggestion

Make sure you follow the submission instructions from each project page. For this project you need to submit two files

- jupyter notebook (finding_donors.ipynb)
- html version of the notebook (renamed it as report.html)


Machine Learning Engineer Na
https://classroom.udacity.com/nanodegrees/nd009t/parts/55674913-b76
Search
Project:
Finding Donors for CharityML
1. Overview
2. Software Requirements
3. Starting the project
4. Submitting the project
5. Project Workspace
6. Project: Finding Donors for Charity...
Submitting the project
NEXT
ENG
15:59
03-03-2018

criteria found in the rubric must be *meeting specifications* for you to pass.

Submission Files

When you are ready to submit your project, collect the following files and compress them into a single archive for upload. Alternatively, you may supply the following files on your GitHub Repo in a folder named `finding_donors` for ease of access:

- The `finding_donors.ipynb` notebook file with all questions answered and all code cells executed and displaying output.
- An **HTML** export of the project notebook with the name **report.html**. This file *must* be present for your project to be evaluated.

Using your workspace(in the next node) you can submit your project directly for review. Remember to upload the HTML report once you are done working on the Jupyter notebook into your workspace BEFORE submitting your report.

Once you have collected these files and reviewed the project rubric, proceed to the project submission page.

Make sure you submit both files on next submission.

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

Preparing the Data

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

Evaluating Model Performance

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

You correctly calculated both accuracy and f-score. Good work!

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.

Nice explanation of each of selected models pros, cons and reason of selection.

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

Required changes

It seems the previous reviewer forgot to point out that you passed parameters to accuracy_score in wrong order.

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

Please re-run this section after you correctly implement the pipeline code.

Improving Results

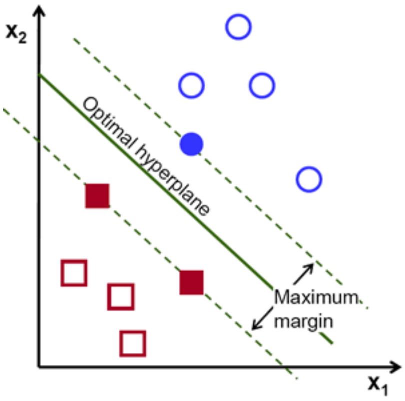
Justification is provided for which model appears to be the best to use given computational cost, model performance, and the characteristics of the data.

I agree with your final model selection but please upate the description as per result graphs after you re-run the models with pipeline implementation.

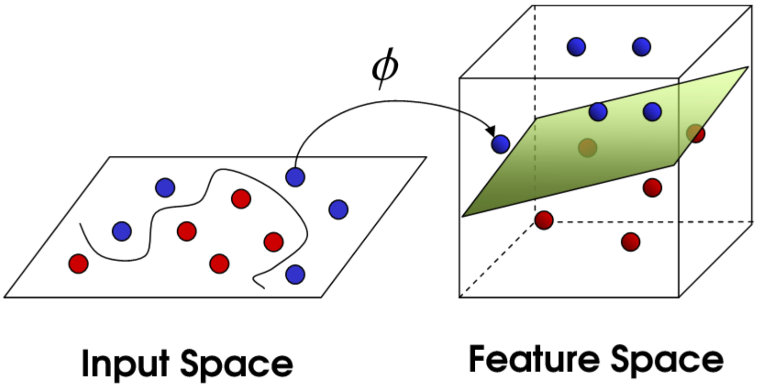
Student is able to clearly and concisely describe how the optimal model works in layman's terms to someone who is not familiar with machine learning nor has a technical background.

The explanation is still not in layman's terms. Here is a sample explanation of svm in layman's terms :

The SVM takes data about individual (age, workclass, education_level, etc), and uses them to create a function that draws a boundary between individual makes more than $50,000 and those who did not. The boundary between the two parameters(>$50,000 or <=$50,000) can be maximized, so that it perfectly classify the data.



Often, though, it's not easy to draw a decision boundary in low dimensions, so the SVM separates the individual makes more than $50,000 and those who did not by projecting the data into higher dimensions. So that the data can be easily separable.



Using this function created with individual makes more than $50,000 and those who did not, the SVM can look at new individual's data and predict accurately.

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.

Nice implementation using grid search!

You could actually go well beyond grid search and implement ‘pipelines’ where the whole machine learning process becomes 'grid-searchable' and you can parameterize and search the whole process though cross validation.
http://scikit-learn.org/stable/modules/generated/sklearn.pipeline.Pipeline.html
And yes you can try out several algorithms automatically as well too! Watch out though this is pretty advanced stuff, here is a great, informative, top notch tutorial from Zac Sewart!
http://zacstewart.com/2014/08/05/pipelines-of-featureunions-of-pipelines.html

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.

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.

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.

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.

RESUBMIT PROJECT

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