UDACITY

‹ Return to "Machine Learning Engineer Nanodegree" in the classroom

DISCUSS ON STUDENT HUB

Finding Donors for CharityML

REVIEW

CODE REVIEW

HISTORY

Meets Specifications

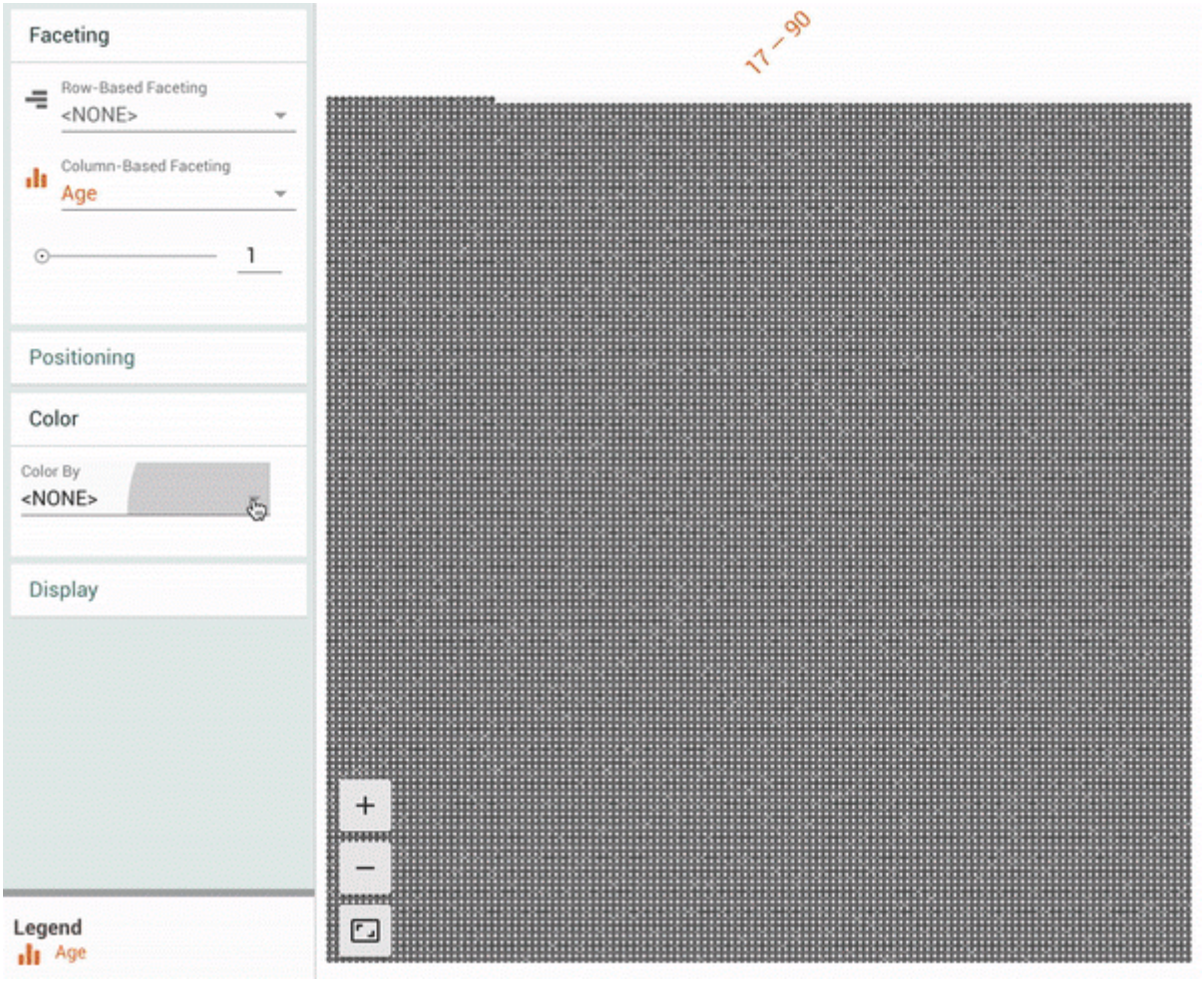
Dear Student,

Overall, well done!. You demonstrate a good understanding of Machine Learning concepts and your responses are written in explanatory terms allowing your audience to understand the work done. 💪

Congratulations on passing your exam!

GREAT NEWS!!

A partnership with Google recently released an open source software to visualize machine learning datasets. And the best part is that they are using the same dataset of this project as live demo! If you want to perform some data visualization with the data of this project, go to the link and play around. Here's an example from Google's blog:



PD: *"If you can focus on the explanation of Gradient Boosting algorithm then that would be great."*. The given explanation is great, good work! 💪

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**

Nice job getting these numbers.

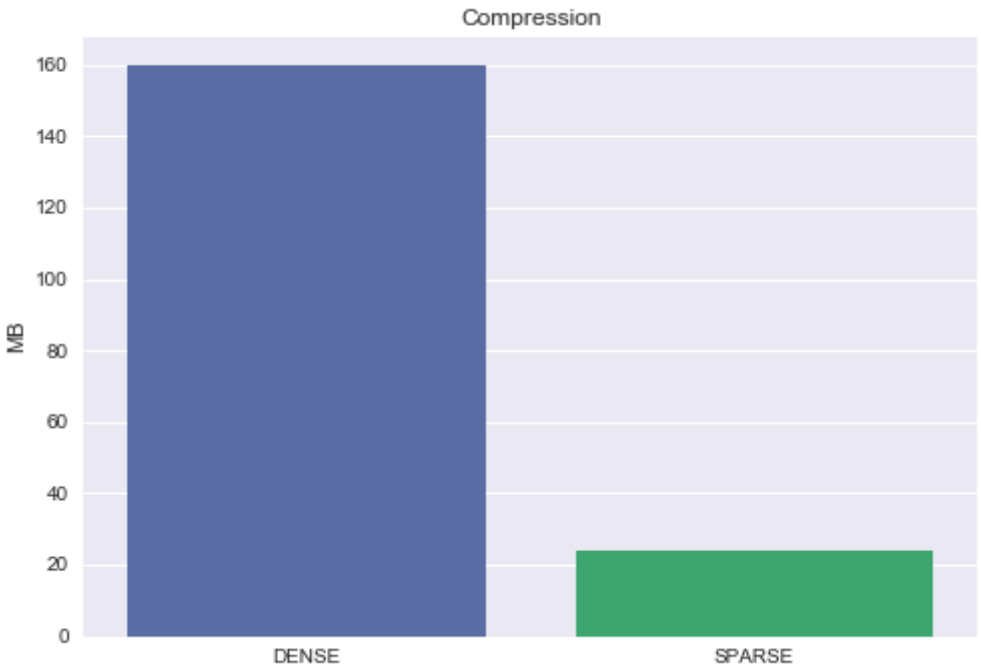
Preparing the Data

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

Excellent implementation of one-hot encoding.

For your reference, this is a great post that demonstrates different ways to handle categorical variables.

Also, when you work with huge datasets, you might benefit of using Sparse matrices which are a memory efficient way to store matrices that are almost 99% zeroes and 1% ones.



Evaluating Model Performance

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

Well done getting the right expression for the F-score. Note in this case, accuracy and prediction are the same!.

- Accuracy: the percent of times the model is right
- Precision: it is the percent of times the model is right on its "more than $50,000" predictions

Since the model is a naive predictor an always predicts "more than $50,000", both scores are similar! 😉

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.

Great discussion of the pros and cons for the different algorithms attempted, as well as their main applications and reasons to use them in this problem.

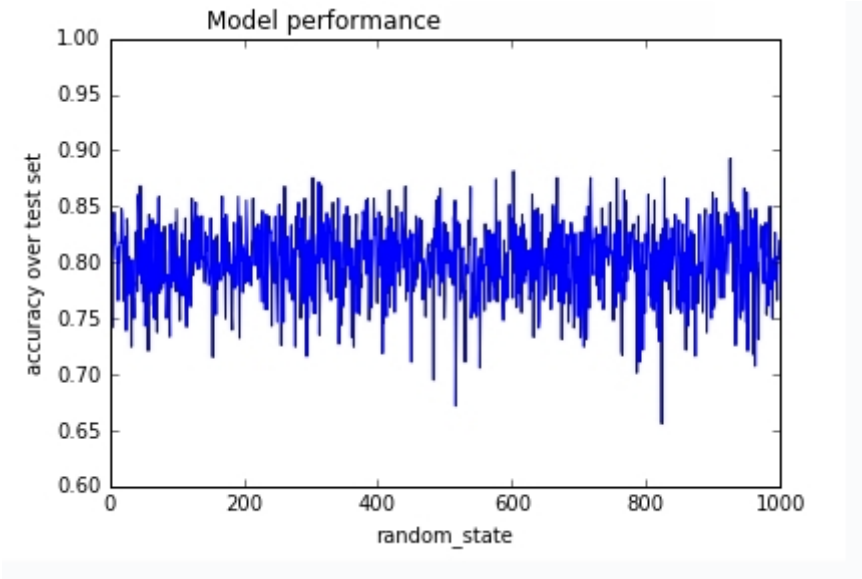
As a side comment, you choose a great stack of models to attack your problem!. In my opinion, GaussianNB is always a good choice since it doesn't require tuning and can reach decent results even if features are not fully independent, it could be your benchmark to explore further algorithms. Decision Trees are also a good choice since they provide feature importances which can help you to understand the predictive power associated with each feature and also an intuitive visualization of the decision criteria. Another right choice would be Logistic regression since there are just two classes, this classifier is quite robust and also provides probability estimates so you can understand the model confidence when labeling a student with a particular class. For more accurate results, but computationally more expensive, ensemble models are the right choice, even you can consider a Voting Classifier to combine the output of several classifiers!. As you can see, there are plenty of options!, it is a matter of the dataset characteristics and your main goals to achieve to select the right algorithms.

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

Excellent implementation of the pipeline to train and validate the different classifiers attempted.

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

Well done testing the different classifiers attempted for different training sizes as well as setting the `random_state` for those algorithms that call for it. Note `random_state` affects the model performance, so for reproducibility purposes, it is required to define a particular value. In the example below, it is tested the same algorithm but using values for `random_state` in the interval `[0, 1000]`:



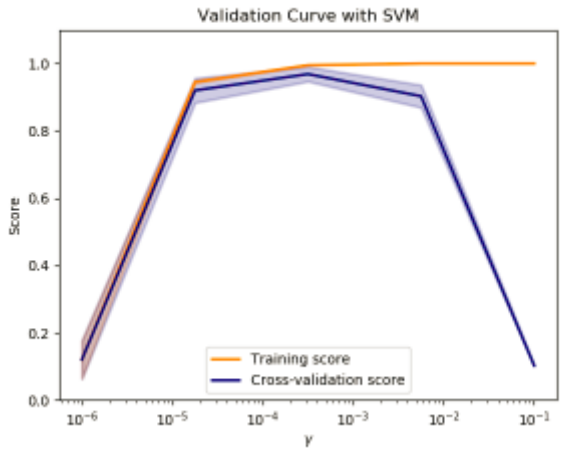
Improving Results

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

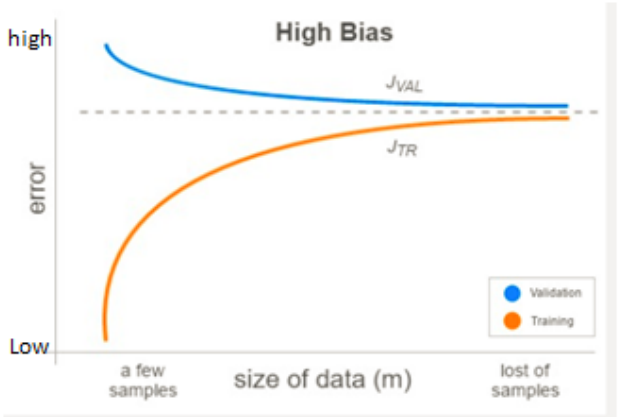
Nice work using your results, in terms of performance and computational cost, to justify the final choice.

As a side comment, Scikit includes a whole chapter on validation.

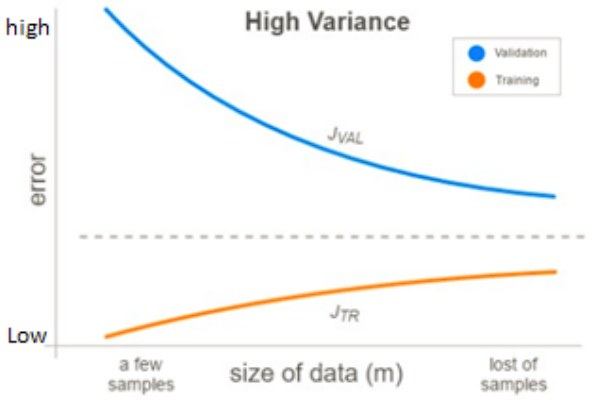
Validation curves are used when it is helpful to plot the influence of a single hyperparameter on the training score and the validation score to find out whether the estimator is overfitting or underfitting for some hyperparameter values.



On the other hand, learning plots are used to find whether our model is overfitted or underfitted. Basically, this is something that we did in this exercise by comparing the train versus test sets performances, if the performance on the train set is high but our test set score is significantly lower, the model is **overfitted**, that is, with the aim to maximize the performance on the train set, the model identified during the training phase those patterns that are particular just to this set and are not replicable on other sets (remember, a dataset is always an **approximation** of the whole population and is always biased). This scenario is a huge problem since the model learned these particularities that are just present in the train set and not those patterns that could make it generalize properly.



Learning Curve in a High Bias model



Learning Curve in a High Variance model

For more insights, check this great post on Learning curves.

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.

Excellent work describing how the algorithm tries to separate people with more/less than $50K using the information obtained from several agents.

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.

Well done tuning your algorithm with SearchGridCV and obtaining the best estimator from it. As a suggestion, using the `cv` parameter you can pass a cross-validation object to validate your search results that best adapt to your dataset characteristics (unbalanced and small).

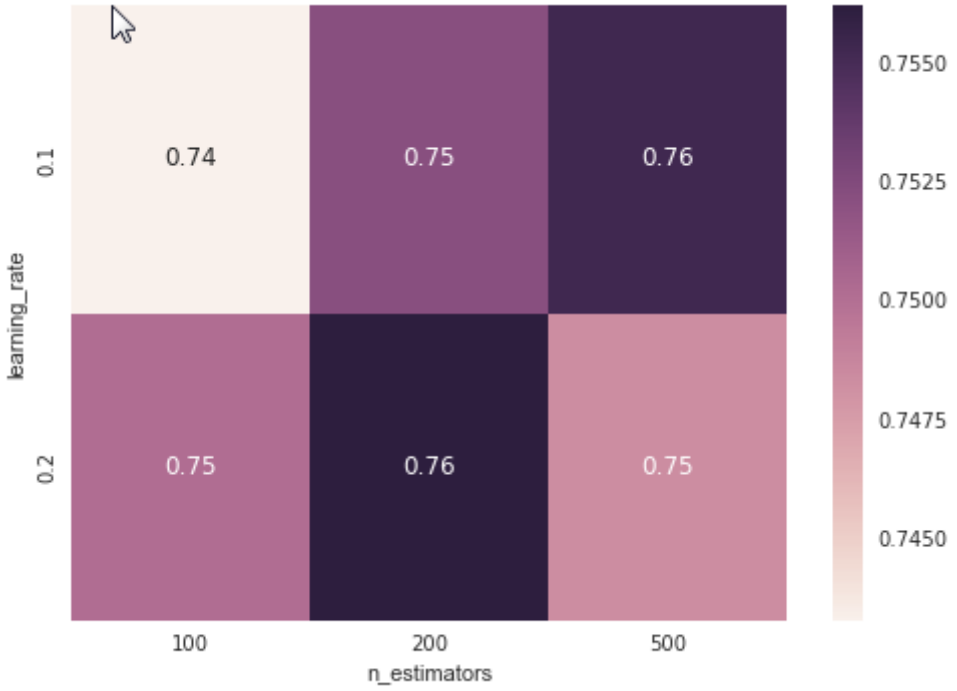
For example:

```
# Set up cross validator (will be used for tuning all classifiers)
    from sklearn import cross_validation
    cv = cross_validation.StratifiedShuffleSplit(labels, 100, random_state =
42)
    a_grid_search = GridSearchCV(clf, param_grid = clf_params,cv = cv, scorin
g = 'recall')
```

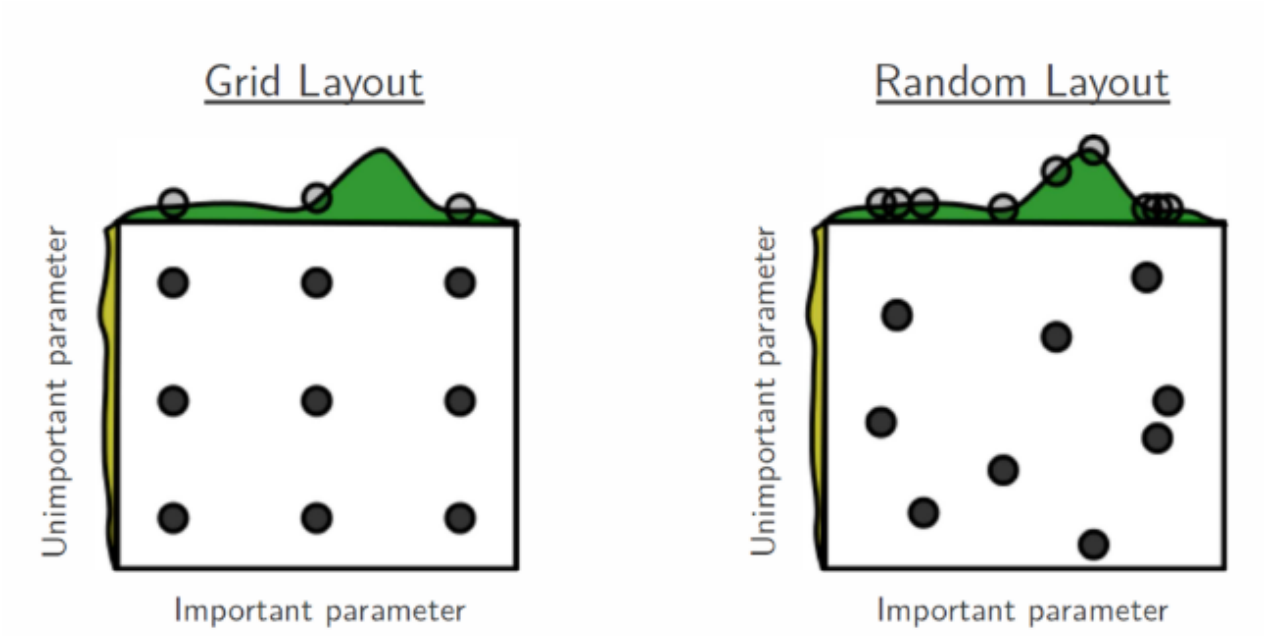
For your reference, note you can also visualize your grid results. For example:

```
# checking how well did we perform !
gridResults = grid_fit.grid_scores_
gridResultsDf = pd.DataFrame([[r[0]['n_estimators'],r[0]['learning_rate'],r
[1]] for r in gridResults],columns=
                             ['n_estimators','learning_rate','score'])

import seaborn as sns
sns.heatmap(gridResultsDf.pivot(columns='n_estimators',index='learning_rate',
values='score'), annot=True)
```



Finally, note a more efficient method to optimize parameter is the RandomSearch. Please check this interesting post on this topic:



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.

Good discussion of the performances obtained along the whole process.

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.

Well done summarizing those features that intuitively seem important, it seems individual's income level is a key factor.

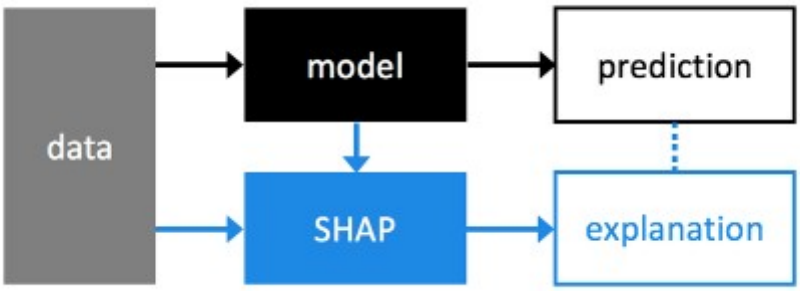
Student correctly implements a supervised learning model that makes use of the `feature_importances_` attribute. Additionally, student discusses the differences or similarities between the features they considered relevant and the reported relevant features.

Well done getting the feature importances and good intuition, you were mostly right. 😎

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.

Good analysis. Note feature selection process is key in Machine Learning problems, the idea behind it is that you want to have the minimum number of features that capture trends and patterns in your data. A good feature set contains features that are highly correlated with the class, yet uncorrelated with each other. Your machine learning algorithm is just going to be as good as the features you put into it. For that reason, this is definitely a critical step in any ML problem. In this case, there is no gain in terms of performance, but in terms of computational costs and model explicability, there is a significant gain!.

BTW, here more info on Shap.



DOWNLOAD PROJECT

RETURN TO PATH

Rate this review