UDACITY

< Back to Machine Learning Engineer Nanodegree

Machine Learning Capstone Project

REVIEW

CODE REVIEW

HISTORY

Meets Specifications

This is a very cool analysis and a great read to a very real world and practical problem. You have demonstrated a full understanding of the entire machine learning pipeline and your report definitely gets the readers attention with the results you have achieved.

Hopefully you have learned a bunch throughout this capstone project(as I can image that you have by reading your report) and you can take some of these techniques further.

If this is your final report, I would like to be the first one to congratulate you on completing this nano-degree! Wish you the best of luck in your future!

Definition

Student provides a high-level overview of the project in layman's terms. Background information such as the problem domain, the project origin, and related data sets or input data is given.

Nice work here with your opening section, as you have given good starting paragraphs to outline the project and have provided background information on the problem domain. Definitely a real world problem.

And you have provided good research to back your claims. It is always important to provide similiar research on such a topic.

The problem which needs to be solved is clearly defined. A strategy for solving the problem, including discussion of the expected solution, has been made.

> "This project aims to discover a classification model that can accurately predict the state of Kickstarter projects (success or failure) given the project characteristics. "

Problem statement is clearly defined here. And glad that you mention that this would be a classification problem in this section.

And very nice job mentioning your machine learning pipeline here, as this gives the reader some ideas in what is to come in your report and how you plan on solving this important task.

Metrics used to measure performance of a model or result are clearly defined. Metrics are justified based on the characteristics of the problem.

Awesome analysis and justification for your F1 Score metric.

> "In situations where the distribution of the target variable in a binary or multiclass classification task is unbalanced, the accuracy score becomes a less reliable indicator of the performance of a model. "

Good justification for your choice in your F1 Score metric. As using accuracy is not the best for unbalanced datasets based on the accuracy paradox (https://www.youtube.com/watch?v=2akd6uwtowc&list=PL6397E4B26D00A269&index=30).

Analysis

If a dataset is present, features and calculated statistics relevant to the problem have been reported and discussed, along with a sampling of the data. In lieu of a dataset, a thorough description of the input space or input data has been made. Abnormalities or characteristics about the data or input that need to be addressed have been identified.

Excellent description of your dataset here, as it is clear in what you will be working with throughout this project.

Would suggest also showing a sample of your dataset (a `data.head()` will do). As this allows the reader to get an understanding of the structure of the data you are working with.

Maybe also look into computing the Kolmogorov-Smirnov test for goodness of fit. (https://docs.scipy.org/doc/scipy-0.14.0/reference/generated/scipy.stats.kstest.html)

Need any data transformations for the features?

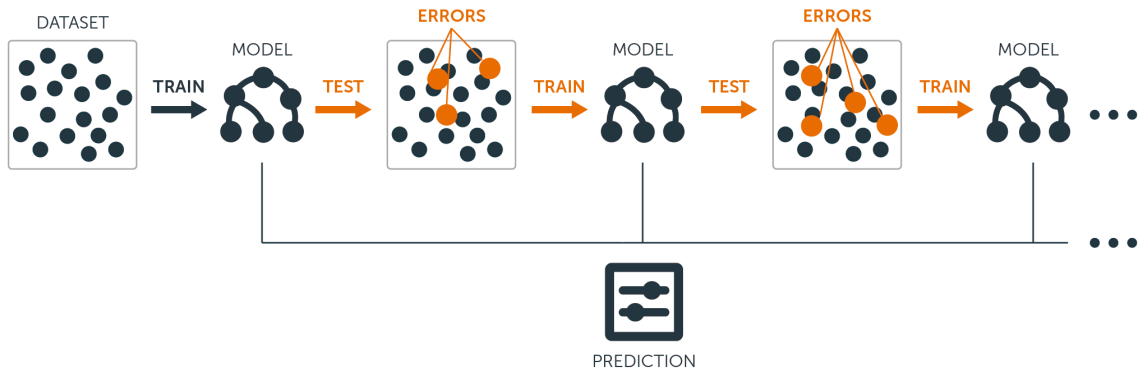
A visualization has been provided that summarizes or extracts a relevant characteristic or feature about the dataset or input data with thorough discussion. Visual cues are clearly defined.

Plotting the Successful and Failed Projects by Main Category is a good idea.

Maybe another idea would be to combine the target variable with these features to get a better understanding of how each correlate to the target variable with a seaborn factorplot.

Algorithms and techniques used in the project are thoroughly discussed and properly justified based on the characteristics of the problem.

Excellent job describing your main models here, as it is very clear that you have a solid understanding in how these models work. Maybe even some visuals could help explain these as well.



Another idea to think about would be if an interpretable model would be more suitable for this prediction problem. As this is always something we need to think about in practice.

Student clearly defines a benchmark result or threshold for comparing performances of solutions obtained.

> "The benchmark to which I will compare the performance of the above three classifiers is a logistic regression model. Logistic regression is one of the simplest but still most widely used classifiers in ML, especially for smaller, less complex datasets. "

Good choice in a benchmark model here. Always good to get a baseline score for your dataset with a linear model.

Benchmarking is the process of comparing your result to existing method or running a very simple machine learning model, just to confirm that your problem is actually 'solvable'.

Methodology

All preprocessing steps have been clearly documented. Abnormalities or characteristics about the data or input that needed to be addressed have been corrected. If no data preprocessing is necessary, it has been clearly justified.

Excellent job documenting all your pre-processing steps. Especially with an 'interpretable' dataset like this one, understanding your dataset, creating features, and important pre-processing steps are always needed!

Nice job mentioning how you dealt with missing values. If you do want to get fancy, you could also run a supervised learning model to 'predict' the Nan value!

The process for which metrics, algorithms, and techniques were implemented with the given datasets or input data has been thoroughly documented. Complications that occurred during the coding process are discussed.

Very solid step by step process here, as it is quite clear in how you approached this problem. Your results would definitely be replicable.

If you would like to learn about some more advanced techniques and combining gridSearch with other techniques, with the notion of Pipelining, check out this blog post brought to you by Katie from lectures

- (https://civisanalytics.com/blog/data-science/2016/01/06/workflows-python-using-pipeline-gridsearchcv-for-compact-code/)

The process of improving upon the algorithms and techniques used is clearly documented. Both the initial and final solutions are reported, along with intermediate solutions, if necessary.

Nice work with your gridSearch ideas, as this is a great way to improve your model. And you have made it very clear in the parameter you tried and the results.

Maybe one other idea, since you have built multiple models, would be to 'ensemble' all of them. As we can typically 'squeeze out' a few more percentage points by doing so. Could look into using VotingClassifier

Results

The final model's qualities — such as parameters — are evaluated in detail. Some type of analysis is used to validate the robustness of the model's solution.

You have good analysis of your final models.

Would suggest expanding in your analysis to validate the robustness of the model's solution. Think about ways to validate your models?

Few ideas to validate your one final model

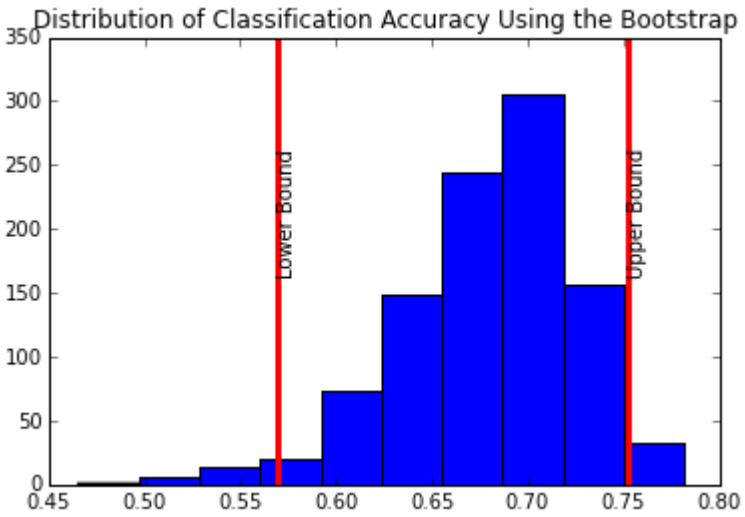
- Could look into using KFold CV and show the different folds scores
- Run your model with multiple different random states and show the mean / variance of the results
- What about small changes in the dataset will this affect this model?

Maybe one other idea would be to plot and 95% confidence interval to determine if the model is robust with bootstrapping. Here might be an example with the boston housing dataset.

```
from sklearn.utils import resample
import matplotlib.pyplot as plt

data = pd.read_csv('housing.csv')
values = data.values
# configure bootstrap
n_iterations = 1000
n_size = int(len(data) * 0.50)

# run bootstrap
stats = []
for i in range(n_iterations):
    # prepare train and test sets
    train = resample(values, n_samples=n_size)
    test = np.array([x for x in values if x.tolist() not in train.tolist
()])
    # model
    model = DecisionTreeRegressor(random_state=100)
    model.fit(train[:,:-1], train[:,-1])
    score = performance_metric(test[:,-1], model.predict(test[:,:-1]))
    stats.append(score)

# confidence intervals
alpha = 0.95
p = ((1.0 - alpha) / 2.0) * 100
lower = max(0.0, np.percentile(stats, p))
p = (alpha + ((1.0 - alpha) / 2.0)) * 100
upper = min(1.0, np.percentile(stats, p))

# plot
plt.hist(stats)
plt.axvline(lower, color='red', lw=3)
plt.text(lower, n_iterations // 4, 'Lower Bound', rotation=90)
plt.axvline(upper, color='red', lw=3)
plt.text(upper, n_iterations // 4, 'Upper Bound', rotation=90)
```

```
plt.title('Distribution of Classification Accuracy Using the Bootstrap')
plt.show()

print('%.1f confidence interval %.1f%% and %.1f%%' % (alpha*100, lower*100
, upper*100))
```

(https://machinelearningmastery.com/calculate-bootstrap-confidence-intervals-machine-learning-results-python/)



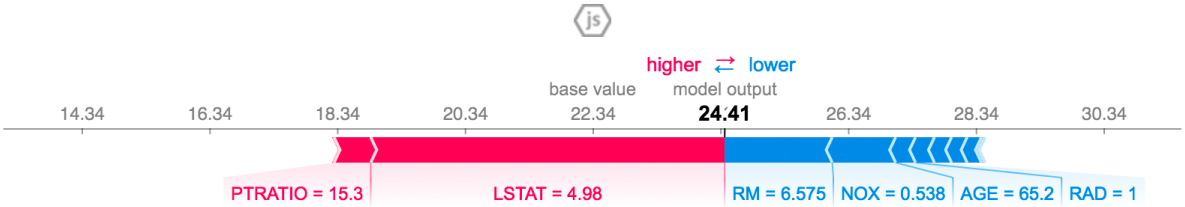
The final results are compared to the benchmark result or threshold with some type of statistical analysis. Justification is made as to whether the final model and solution is significant enough to have adequately solved the problem.

Conclusion

A visualization has been provided that emphasizes an important quality about the project with thorough discussion. Visual cues are clearly defined.

Feature Importance plots are always a good idea to determine the most important features for the predictions. This also can help the company determine where they need to focus on as well.

Another really cool idea would be to check out the SHAP library. SHAP (SHapley Additive exPlanations) is a unified approach to explain the output of any machine learning model. SHAP connects game theory with local explanations, uniting several previous methods and representing the only possible consistent and locally accurate additive feature attribution method based on expectations (see the SHAP NIPS paper for details). This is where you can visualize your machine learning model's predictions with visuals such as



Student adequately summarizes the end-to-end problem solution and discusses one or two particular aspects of the project they found interesting or difficult.

Nice work discussing your final end-to-end problem solution as this reads quite well. I can definitely tell that you have spent a long time on this project as it really shows.

Discussion is made as to how one aspect of the implementation could be improved. Potential solutions resulting from these improvements are considered and compared/contrasted to the current solution.

> "Similarly, a multi- layer perceptron algorithm may have been well-suited to this task, though I did not explore this avenue."

Maybe even try and use a mixed input NN model. Check out this paper (https://arxiv.org/abs/1604.06737)

And here might be an idea of how this can be done in Keras (https://github.com/entron/entity-embedding-rossmann)

Quality

Project report follows a well-organized structure and would be readily understood by its intended audience. Each section is written in a clear, concise and specific manner. Few grammatical and spelling mistakes are present. All resources used to complete the project are cited and referenced.

Your writing is very clean and it is very easy to understand what you are saying. I personally thank you as this report is very easy to read :)

Code is formatted neatly with comments that effectively explain complex implementations. Output produces similar results and solutions as to those discussed in the project.

Code does look good.

DOWNLOAD PROJECT

Student FAQ