UDACITY

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

DISCUSS ON STUDENT HUB

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 is essentially an exercise for which objective numerical inputs are “regressed” against subjective binary quality outputs. Here, an arbitrary (sensory output variable) quality value of 7 would be utilised, above which the wine is classified as good quality and below which, poor quality."

Problem statement is clearly defined here, and glad that you mention that this is 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 metrics!!

Tying your metric choice into your particular problem and problem domain is actually the single most important thing to do in any machine learning project. If you optimize a model based on the incorrect metric, your model might not be suitable for the business goals.

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.

Very nice job describing your dataset. Glad that you show some descriptive stats, show a sample of your data, go into a bit of detail in the features here and the distribution of the target variable. 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.

All great visuals to show here! The distribution of target variable and correlation plots are always 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

You might also check out using some more advanced plotting libraries such as

- plot.ly: Modern Visualization for the Data Era. Where you can create really cool interactive visuals in jupyter notebooks and web apps!

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. Really like the visuals to help explain these as well.

Another idea to think about would be if an interpretable model would be more suitable for predicting wine quality. Would your DecisionTreeClassifier be better, as we can actually determine the features that correlate to the prediction? Or is the most predictable model best? 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.

> "There is some differences however, since the original paper retains the scale of 0-10 for the classification task (a multiclass classification), whereas in this proposal a binary classification task with an arbitrary cut-off quality value of 7 has been proposed instead."

If this is the case, this shouldn't be used as a benchmark then. Your benchmark model needs to be run on **the exact same dataset** that you are using, so your benchmark results are comparable to your 'final optimized' model's results.

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.

Min-Max Scaling is a fine idea. You might also look into using RobustScaler. This Scaler removes the median and scales the data according to the quantile range (defaults to IQR: Interquartile Range). The IQR is the range between the 1st quartile (25th quantile) and the 3rd quartile (75th quantile).

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.

To make your implementation very smooth, might want to play around with some more advanced techniques, such as combining scaling, fitting, gridSearch, etc... 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 parameters you tried and the results.

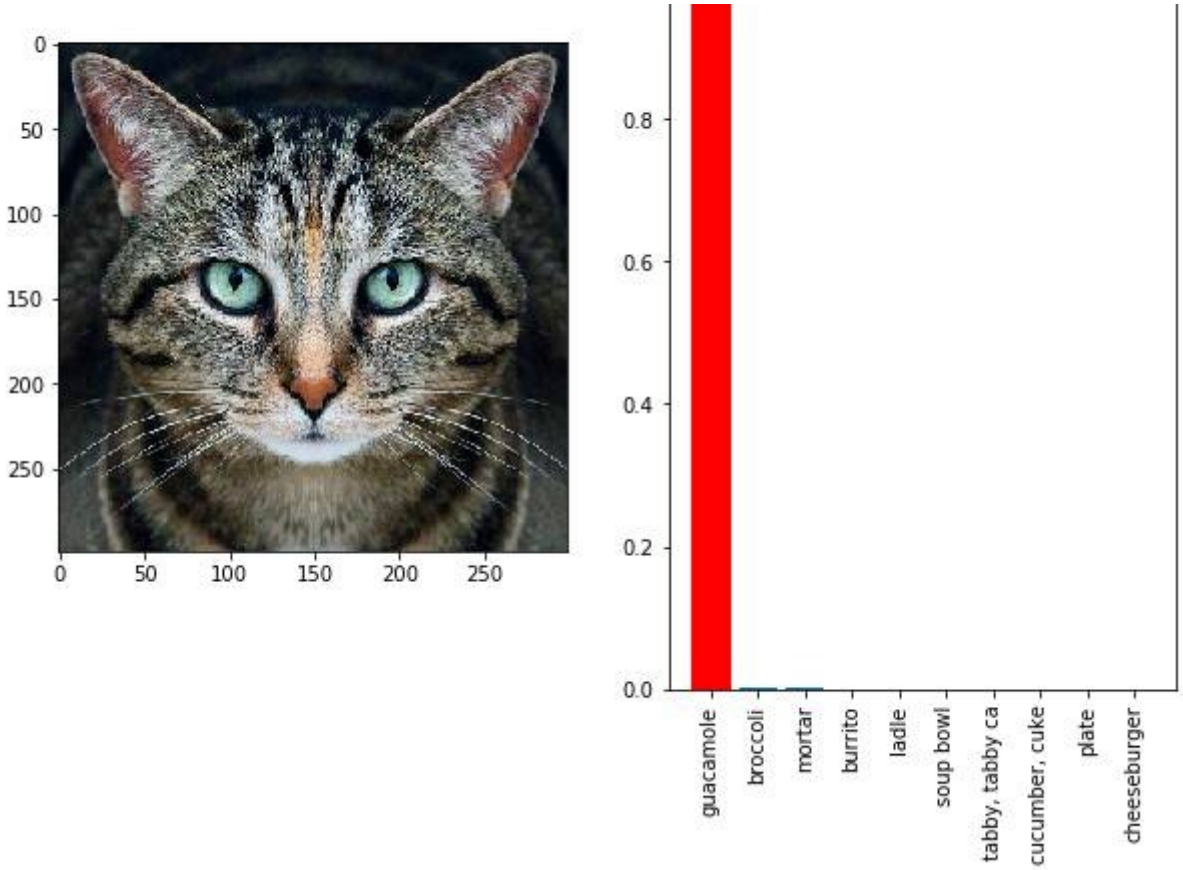
Maybe one other idea, since you have built multiple models, would be to 'ensemble' all of them. As we can typically 'squeeze out' a few more percentage points by doing so. Could look into using VotingClassifier

Results

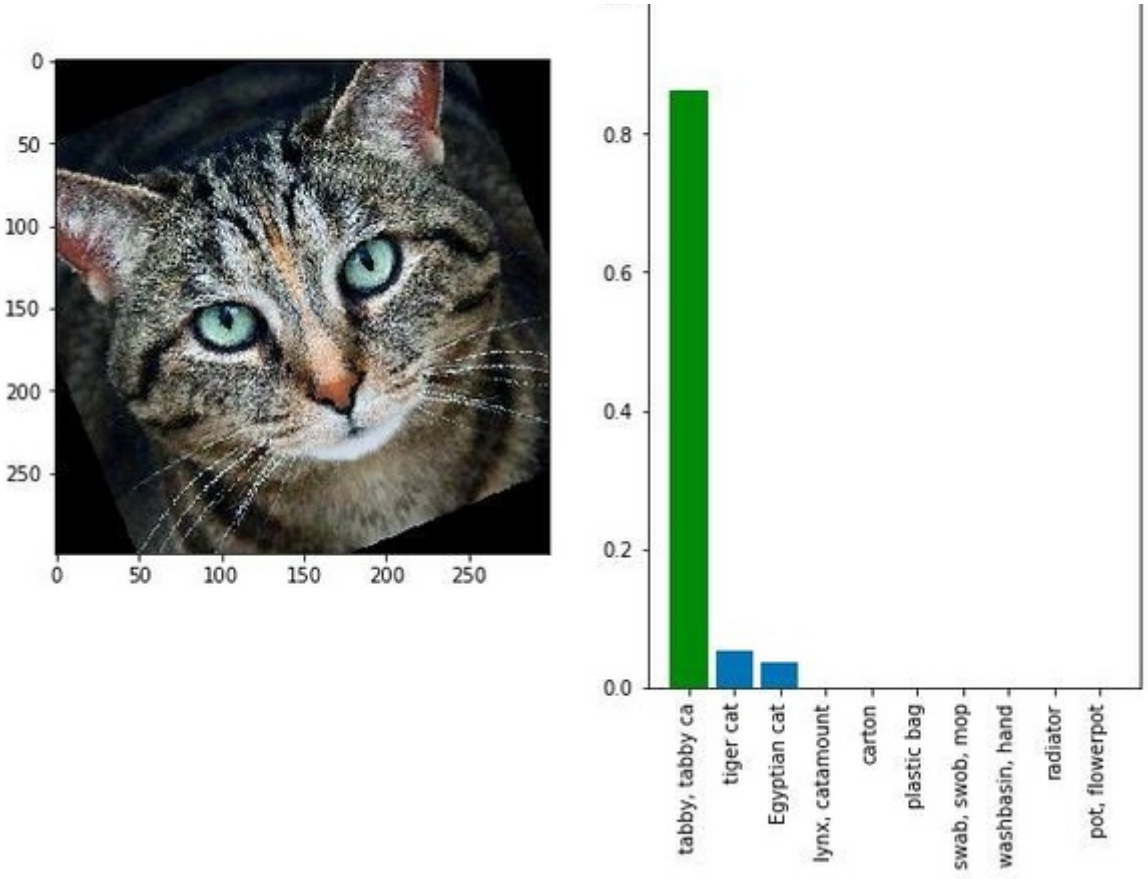
The final model's qualities — such as parameters — are evaluated in detail. Some type of analysis is used to validate the robustness of the model's solution.

You have good analysis of your final models and excellent analysis to validate the robustness of the model's solution by running your model with different random states. Great analysis and table!

Why we need to validate our models?



Here's a picture of a cat right? Google's Inception model thinks it's a guacamole. As much as the image looks like a cat, the image is digitally altered which confused the model.



Slightly rotating the image led the model to correctly classify the image as a cat (and as an animal)

The above image is what's called as an adversarial image, trying to fool your model into thinking the image is something you want it to be instead of what the image actually is.

The real danger is in the application, especially in healthcare and defense. For example, how would you convince that your model for predicting cancer actually works? How do you know your model is not susceptible to noise? How do you know that your model has actually learnt what it is supposed to be learning? This is why it is always important to validate the robustness of the model's solution.

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.

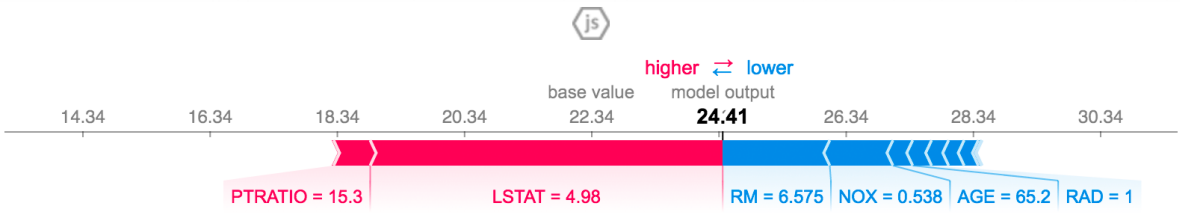
Conclusion

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.

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.

Looks good! You might also check out these links for some best practices when commenting your code

- this post regarding Docstrings vs Comments.
- Google Style Python Docstrings
- This Best of the Best Practices" (BOBP) guide to developing in Python

DOWNLOAD PROJECT

RETURN TO PATH