Project 1

Due date: Monday, April 1, 2024, 11:59 PM

Objectives

- Effectively function as a member of a team to apply fundamental concepts of artificial intelligence to solve real-world problems.
- Develop and rigorously evaluate multiple predictive models for a specified real-world problem, adhering to industry best practices throughout the process.
- Perform data cleaning, feature engineering, and feature selection, as necessary, to prepare the dataset for effective model training.
- Train various predictive algorithms and meta-algorithms and conduct grid searches within a cross-validation procedure to identify a good configuration, ensuring the evaluation is robust and reliable.
- Determine the most effective model and explain the project's findings.
- Communicate technical results clearly and correctly, in writing and verbally.

Overview

This is a team project, and the amount of effort devoted should be proportional to the number of people in the team, and all team members should make an equal contribution. You will have to present your results, findings, and interpretations to your classmates. Each team member must be active in the presentation. You must also evaluate the other teams' presentations and assess the correctness of their work. You will also be graded on your evaluation of the other presentations and the rigor and correctness of the other projects. You will be required to attend class on 4/02 and 4/04. The schedule for presenting is available in Moodle.

Tasks

I. Select your project

1. Forecast the 2024 College Basketball Tournaments
2. House Selling Prices
3. Titanic - Machine Learning from Disaster
4. Spaceship Titanic - Which passengers are transported to an alternate dimension?
5. Emotions: Where Words Paint the Colors of Feelings

Note: If you want to work on a different project, you must consult with me no later than Thursday, 3/7. The project must be an ongoing competition on the Kaggle platform.

II. Prepare data for training

1. Download the data (and format it) as a CSV file, enabling it to be easily imported into a pandas DataFrame for preliminary analysis and subsequently supplied to a learning algorithm.

In your presentation, you must include the following details (2.- 3.) about the original dataset. Throughout this document, text in blue indicates the information that needs to be included on the slides of your presentation.
It helps to be clear on the dataset upfront, so now is the time to gather this information.

2. What is the prediction problem?
   - Be prepared to clearly define the problem and deliver a concise elevator pitch about what you're aiming to solve. For instance, you might say, "We're working on predicting a house's selling price by considering factors like its size in square feet, the year it was built, and its location within the city."
   - Also, specify the type of machine learning task you're tackling, whether a regression or a classification problem.
3. How big is the dataset?
   - **size on disk** (in MB or GB, *etc*.) Depending on the algorithm and number of features and instances, training a model in a cross-validation procedure may be quite time-consuming; you may consider utilizing only a subsample of the training points. Keep track of how long training takes.
   - **number of training instances m** (typically the number of rows in the file, without the header, of course). If working with a sample, specify the original size and the sample size.
   - **number of features n** (typically the number of columns excluding the label, aka dimensionality of the problem)
   - **range** (min value and max value) of the **numeric label** (if you are doing regression) or **number of classes** and **class distribution** (i.e., number of instances per class, if you are doing classification)
4. Set aside 20% of the data (or the sample) into a new file called ***test.csv***, and DO NOT touch it until instructed to do so. If you are working on classification, make sure the split is done in a stratified manner to maintain class priors/distribution, otherwise it will be difficult to make predictions. Note: For regression, keeping the numeric values properly binned requires slightly more work. (May skip.)
5. The remaining 80% of your initial dataset is referred to as the development set (dev.csv). This is what you will use in cross-validation to build your models.

III. K-Fold: Establish a baseline performance metric

1. Utilize the dev set, incorporating all its attributes, and perform stratified K-fold cross-validation with either k = 5 or k = 10.
   - For regression problems, train a linear regression model using default hyperparameter settings and calculate the Mean Squared Error (MSE) or another relevant metric (or, as specified by Kaggle requirements).

- For classification problems, implement logistic regression with default hyperparameters and determine the accuracy or an alternate metric, (or, as specified by Kaggle requirements).
- Analyze the performance resulting from the K-fold cross-validation. This is to get a general idea about the quality of the data.

2. Average the performances of the folds and determine one number (e.g., accuracy = 0.7 or MSE = 195.2). This initial assessment is the baseline you must improve upon.

IV. K-Fold: Algorithm training and continuous evaluation

3. In the same k-fold ***stratified*** cross-validation procedure from III, perform a grid search of the best algorithm and its corresponding hyperparameters, either manually by trying out different hyperparameter values or by using `sklearn`'s grid search functionality. The algorithms to try, in addition to (logistic) regression, are:
   A. Decision Tree
   B. Random Forest
   C. Naive Bayes algorithm (if doing classification)
   D. One additional algorithm you choose, different from the ones above, can be MultiLayer Perceptron, XGBoost, Neural Networks, or any other algorithm that has not been covered yet. Be ready to give some intuition about it during the presentation.
4. For each of these algorithms, try different hyperparameter values (at least five per algorithm; e.g., use different tree depths for your decision trees) and different attribute subsets. Track the iterations of your models to easily enable rollback and comparisons. Make sure you use the exact same splits – by setting the random state to `3270`, for any random number generator used – to objectively compare *configurations*.

We will refer to `features + algorithm + hyperparameters`, i.e., a (sub)set of features, an algorithm, and its hyperparameter values, as *one configuration*. You will need to identify which configurations significantly enhance the predictive capabilities of your dataset. It is up to you to define what constitutes a "significant" improvement and provide a rationale for your conclusion.

5. Once you have at least 25 models (corresponding to 25 configurations) evaluated in k-fold cross-validation, identify the top three best performers: c1, c2, and c3 among all configurations.
6. Using these top configurations, train three new models using the entire dev dataset, not in a CV procedure but in one single go. Then, evaluate these three models on the test dataset you set aside in II-4. This will represent the predictive accuracy closest to the reality of your three best models, and what is the expected performance in a real-world production setting. Show the top three configurations, more specifically show their cross-validation results and their results on the final test dataset.

7. Which configuration emerged as the top performer, and how do you interpret its success? Here, you can tie the result back to the assumptions made by the algorithm (linear vs non-linear, feature independence, etc.), what attributes you used, over/underfitting, bias-variance trade-offs, etc.
8. Also, include any clever tricks you used to improve the predictive performance, and which trick in particular was most useful (cleaning, scaling, balancing, removing certain features, training meta-algorithms), and why do you think it worked so well?
9. Extra: Now, your most successful configuration can be applied to train on the full dataset mentioned in II-3, make predictions on the test cases provided by Kaggle, and submit the predictions for assessment to get placed on the leaderboard.
10. Extra: What position did you achieve?

Submission

- This is an individual assignment; by submitting it, you certify that no unauthorized help was received or given in completing it.
- Please submit Pylint-standardized code (-1 point for each file not rated 10/10).
- Please name your script(s) `cs3270p1_team(i).py`, your report `cs3270p1_team.pdf`, and your slides `cs3270p1_team.pdf` (-1 point per file if the naming requirement is not satisfied)

Grading

- The max grade is 20 points, equivalent to 20% of the final grade in the course.
    - Tasks (scripts): 8 points
    - Report: 4 points (technical log of your iterations, with brief comments and how decisions were made at various important steps in the process)
    - Presentation: 4 points
    - Evaluation* of other projects: 3 points
    - Evaluation* of your own team: 1 point

* Evaluation forms will be available during the presentation week, in April.

- While grading the project, emphasis will be placed on your understanding of machine learning algorithms and principles, the thoroughness of your approaches, and the correctness of the methodologies you use rather than your final prediction score or your ranking on the competition leaderboard.
- Constructing a robust model is an iterative process where actions must be guided by the evidence (empirical results) from the previous iterations and based on theoretical principles, as opposed to random trial-and-error.
- Bonus: 3 points equivalent to 3% of the course grade

Inspiration

- https://www.kaggle.com/code/prajwalkanade/apple-quality-analysis