EE 559 | **Course Project Assignment** | Post: Sat., 4/9/2022
Jenkins | Due: Mon., 5/2/2022, 11:59 PM

Introduction

The goal of this project is to develop one (or more) machine learning systems that operate on the given real-world dataset(s). In doing so, you will apply tools and techniques that we have covered in class. You may also confront and solve issues that occur in practice and that have not been covered in class. Discussion sessions and past homework problems will provide you with some tips and pointers for this; also internet searches and piazza discussions will likely be helpful.

Projects can be individual (one student working on the project) **or a team** (2 students working together on one project).

You will have significant freedom in designing what you will do for your project. You must cover various topics that are listed below (as "Required Elements"); the methods you use, and the degree of depth you go into each topic, are up to you. And, you are encouraged to do more than just the required elements (or to dive deeper than required on some of the required elements).

Everyone will choose their project topics based on the three topic datasets listed below.

Collaboration and comparing notes on piazza may be helpful, and looking for pertinent information on the internet can be useful. However each student or team is required to do their own work, coding, and write up their own results.

Topic datasets:

There are 3 topic datasets to choose from:

(i) Power consumption of Tetouan City
Problem type: regression

(ii) Student performance (in Portuguese schools)
Problem type: classification or regression

(iii) Algerian forest fires
Problem type: classification

These datasets are described in the appendix, below.

Note: for each topic dataset, you are required to use the training and test sets provided on D2L. Some of the features have been preprocessed (e.g., normalization, transformation, deletion, noise removal or addition). Additionally, we want everyone to use the same training set and test set, so that everyone's system can be compared by the same criteria.

Which topic datasets can you use?

Individual projects should choose any one dataset. If it is the student performance dataset, then pick either the classification or regression version.

Team projects should choose any one dataset. If it is the student performance dataset, then do both classification and regression versions. If it is the forest fire dataset or the power consumption dataset, then you are expected explore the topic in more depth than a typical individual project. For example, this could mean doing more feature engineering, trying more (nonlinear) feature expansion/reduction, or trying more models.

Computer languages and available code

You must use Python as your primary language for the project. You may use Python and its built-in functions, NumPy, scikit-learn, and matplotlib. You may find LibSVM or SVMLight useful, and may use them as well. Additionally, you may use imblearn (imbalanced-learn) functions for undersampling or oversampling of your data if that would be useful for your project; you may use pandas only for reading, writing, and parsing csv files; and you may use a function or class for RBF network implementation (e.g., scipy.interpolate.Rbf). Within these guidelines, you may use any library functions or methods, and you may write your own code, functions, and methods. Please note that for library routines (functions, methods, classes) you use, it is your responsibility to know what the routine does, what its parameters mean, and that you are setting the parameters and options correctly for what you want the routine to do.

Use of C/C++ is generally discouraged (for your own time commitments and because we didn't cover it in relation to ML). However, there could be some valid reasons to use C/C++ for some portions of your code, e.g. for faster runtime. If you want to use it, we recommend you check with the TAs or instructor first.

Be sure to state in your project report what languages and toolboxes/libraries you used; what you coded yourself specifically for this class project; and of course any code you use from other sources must be credited as such.

Required elements

- **The items below give the minimal set of items you are required to include in your project, for each dataset you report on.** Note that you are welcome and encouraged to do more than the minimal required elements (for example, where you are required to use one method, you are welcome to try more than one method and compare the results). Doing more work will increase your workload score, might increase your interpretation score, and might improve your final system's performance.
- **EE 559 content**
    - The majority of your work must use algorithms or methods from EE 559 (covered in any part of the semester).
    - You may also (optionally) try algorithms and methods that were not covered in EE 559 for comparison; and describe the method and results in your report.
- **Consider preprocessing: use if appropriate** [Discussion 7, 8, 9]
    - Tip: you might find it easier to let Python (using pandas) handle csv parsing.

- Normalization or standardization. It is generally good practice to consider these. It is often beneficial if different features have significantly different ranges of values. Normalization or standardization doesn't have to be all features or none; for example, binary variables are typically not standardized.
- For classification problems, if the dataset is significantly unbalanced, then some methods for dealing with that should be included. If it's moderately unbalanced, then it might be worth trying some methods to see if they improve the performance. Some approaches to this are done by preprocessing of the data.
- Representation of categorical (ordinal or cardinal) input data should be considered. Non-binary categorical-valued features usually should be changed to a different representation.

- **Consider feature-space dimensionality adjustment: use if appropriate**
  - You can use a method to try reducing and/or expanding the dimensionality, and to choose a good dimensionality. Use the d.o.f. and constraints as an initial guide on what range of dimensionality to try.
  - In addition to feature-reduction methods we covered in EE 559, feel free to try others (some others are mentioned in the forest fires dataset description).
- **Cross validation or validation**
  - Generally it's best to use cross validation for choosing parameter values, comparing different models or classifiers, and/or for dimensionality adjustment. If you have lots of data and you're limited by computation time, you might instead use validation without cross-validation.
- **Training and prediction**
  - Individual projects should try at least 3 different classification/regression techniques that we have covered (or will cover) in class. Team projects should cover at least 4 classification/regression techniques, at least 3 of which are covered in EE 559. Beyond this, feel free to optionally try other methods.

    Note that the required trivial and baseline systems don't count toward the 3 or 4 required classifiers or regressors, unless substantial additional work is done to optimize it to make it one of your chosen systems.
- **Proper dataset (and subset) usage**
  - Final test set (as given), training set, validation sets, cross validation.
- **Interpretation and analysis**
  - Explain the reasoning behind your approach. For example, how did you decide whether to do normalization and standardization? And if you did use it, what difference did it make in system performance? Can you explain why?
  - Analyze and interpret intermediate results and final results. Especially, if some results seem surprising or weren't what you expected. Can you explain (or hypothesize reasons for) what you observe?
  - (Optional) If you hypothesize a reason, what could you run to verify or refute the hypothesis? Try running it and see.

- (Optional) Consider what would be helpful if one were to collect new data, or collect additional data, to make the prediction problem give better results. Or, what else could be done to potentially improve the performance. Suggest this in your report and justify why it could be helpful.

- **Reference systems and comparison**
  - At least one trivial system and one baseline system are required. Each dataset description states what to use for these systems.
  - Run, and then compare with, the baseline system(s). The goal is to see how much your systems can improve over the baseline's system performance.
  - Also run, and compare with, the trivial system. The trivial system doesn't look at the input values $\underline{x}$ for each prediction; its comparison helps you assess whether your systems have learned anything at all.
- **Performance evaluation**
  - Report on the cross-validation (or validation) performance (mean and standard deviation); it is recommended to use one or more of the required performance measures stated in your dataset's description (in the Appendix, below). If you use other measure(s), justify in your report why.
  - Report the test-set performance of your final (best-model) system.
    - For your final-system test-set performance, you may use the best parameters found from model selection, and re-train your final system using all the training data to get the final weight vector(s).
    - Report on all required performance measures listed in the dataset description.
  - You may also report other measures if you like. Use appropriate measures for your dataset and problem.
- **Written final report and code**
  - Submit by uploading your report and code in the formats specified below, to D2L.

General Tips

1. Be careful to keep your final test set uncorrupted, by using it only to evaluate performance of your final system(s).
2. If you find the computation time too long using the provided dataset(s), you could try the following: check that you are using the routines and code efficiently, consider using other classifiers or regressors, or down-sample the training dataset further to use a smaller N for your most repetitive work. In the latter case, once you have narrowed your options down, you can do some final choices or just your final training using the full (not down-sampled) training dataset.
3. Wherever possible, it can be helpful to consider the degrees of freedom, and number of constraints, as discussed in class. However, this is easier to evaluate for some

classifiers than for others; and yet for others, such as SVM and SVR, it doesn't directly apply.

Grading criteria

Please note that your project will not be graded like a homework assignment. There is no set of problems with pre-defined completion points, and for many aspects of your project there is no one correct answer. The project is open-ended, and the work you do is largely up to you. You will be graded on the following aspects:

- Workload and difficulty of the problem (effort in comparison with required elements, additional work beyond the required minimum, progress in comparison with difficulty of the problem);
- Approach (soundness and quality of work);
- Performance of final system on the provided test set;
- Analysis (understanding and interpretation);
- Quality of written code (per guidelines given in Discussion 6 and related materials posted)
- Final report write-up (clarity, completeness, conciseness).

In each category above, you will get a score, and the weighted sum of scores will be your total project score.

For team projects, both members of a team will usually get the same score. Exceptions are when the quantity and quality of each team member's effort are clearly different.

Final report [detailed guidelines (and template) to be posted later]

In your final report you will describe the work that you have done, including the problem statement, your approach, your results, and your interpretation and understanding.

Note that where you describe the work of others, or include information taken from elsewhere, it must be cited and referenced as such; similarly, any code that is taken from elsewhere must be cited in comments in your code file. Instructions for citing and referencing will be included with the final report instructions. Plagiarism (copying information from elsewhere without crediting the source) of text, figures, or code will cause substantial penalty.

For team projects, each team will submit one final report; the final report must also describe which tasks were done by each team member.

You will submit one pdf of your (or your team's) report, one pdf of all code, and a zip file with all your code as you run it. .

Please note that your **report** should include all relevant information that will allow us to understand what you did and evaluate your work. Information that is in your code but not in your report might be missed. The purpose of turning in your code is so that we can check various other things (e.g., random checks for proper dataset usage, checking for plagiarism, verifying what you coded yourself (as stated in the report), and running the code when needed to check your reported results).

Appendix: Overview of Project Datasets

For all datasets

Use only the provided training and test sets on D2L. In the provided datasets, some of the features may have been preprocessed, and thus are incompatible with the feature representation of the dataset that is posted on the UCI website. You are required to use only the provided (D2L) datasets for your course project work, so that everyone on a given topic is graded on the same datasets.

1. Power Consumption dataset

Description written by: *Fernando*

Summary description

The goal of this problem is to predict the power consumption in three different zones of Tetouan city, Morocco. The measurements were taken every 10-min over an entire year.

Problem type: regression

Data Description

The dataset contains 6 features, 5 of which are numerical, and one is categorical. The numerical features are related to the weather and the categorical feature describes the 10-min interval during which the measurements were taken. In detail:

1. Date Time (date is in month/day/year format)
2. Temperature
3. Humidity
4. Wind Speed
5. General diffuse flows
6. Diffuse flows

Note on training, test, and validation sets

You are provided with a training and a test set. The test set was created by randomly selecting 73 (20%) days out of the 365 available days. To avoid overfitting during the validation or cross-validation procedures, we suggest that you follow a similar approach. In other words, you should create a *day of the year* feature, which will have range $[1, 365]$, and the validation set(s) should be drawn as M random days out of the 292 training days. You might want to use skelarn's GroupKFold class to achieve this more easily. And remember you cannot use the test set to make any decisions about your models!

Further discussion (not required reading for the project): when dealing with time-varying data, we cannot create training and test sets completely at random. We want to avoid the situation where our system makes predictions having knowledge about the future. A realistic approach to this problem is to use the first M days as training and the last 365-M for testing, but this approach would create difficulties when also applied to validation or cross-validation processes.

Therefore, we chose this day-based approach, which presents a good trade-off between the realistic problem (respecting history vs. future) and the original problem (treating every 10-minute period as a separate (independent) data point).

Required performance measures to report

- Coefficient of determination (R-squared, R^2):

$$R^2 = 1 - \frac{\sum_{n=1}^{N}(y_n - \hat{y}_n)^2}{\sum_{n=1}^{N}\left(y_n - \underline{y}\right)^2}$$

  where $\hat{y}_n$ is the predicted value and $\underline{y}$ is the mean value of all y_n

  Note that: $-\infty < R^2 < 1$. Higher values of R^2 indicate models with better performance. Values below zero mean the system performs worse than the trivial system.

  You can code your own implementation or use sklearn's implementation.
- Root Mean Squared Error (RMSE= $(MSE)^{\frac{1}{2}}$). Note that sklearn's implementation can compute either MSE or RMSE based on parameter *squared*

The performance will be evaluated mainly in terms of the R^2 score. However, you must provide values for both R^2 score and RMSE in your report.

Tips

Feature engineering will likely play an important role in this project. You are already required to create one feature (numeric feature from the "date time" feature) before performing any validation procedure, but you are encouraged to create others. Examples: *month*, *weekday or weekend*, *hour of day, minute of day*. You can also consider nonlinear transformations or combinations of features. Note that a nonlinear transformation does not have to be a global nonlinear transformation over all original feature variables; you can be more selective like having a nonlinear transformation over just 1, 2, or 3 variables that you choose, for example: allow a nonlinear function of the minute and/or hour of the day, as a set of new features.

Required reference systems

You must code, evaluate and report the test performance of the following systems.

- **Trivial system:** a system that always outputs the mean of the training data output values.
- **Baseline system:** linear regression using only the 5 numerical features (ignoring date time).

Estimated difficulty level (1-5, 1 is easy, 5 is difficult):

2-3 (easy to moderate)

References

[1] Dataset information: (https://archive.ics.uci.edu/ml/datasets/Power+consumption+of+Tetouan+city)

[2] Relevant paper: https://ieeexplore.ieee.org/document/8703007

2. Student Performance dataset

Description written by: *Shuai Xu*

Summary description

The problem addressed is to predict students' academic performance in secondary schools (and to then use the results to intervene and help at-risk students do better). The data is from students in secondary education of two Portuguese schools. The data attributes include student grades, demographic, social and school related features, and was collected by using school reports and questionnaires. The dataset provided for this project measures the students' performance in the subject of Portuguese language (por).

Problem type: Classification (5-class)* or Regression.

For individual projects, choose either regression or classification to work on.

For team projects, do both regression and classification.

Data Description

The full dataset [1] is posted on the UCI website (includes mathematics and Portuguese topics). For this project we will use the only the Portuguese dataset. The number of input attributes per data point is 30, not counting the grade features (Gx). 13 of the attributes are integer valued, 13 are binary-valued categorical, and 4 are multi-valued categorical. There are no missing values. In detail:

1. school - student's school (binary: 'GP' - Gabriel Pereira or 'MS' - Mousinho da Silveira)
2. sex - student's sex (binary: 'F' - female or 'M' - male)
3. age - student's age (numeric: from 15 to 22)
4. address - student's home address type (binary: 'U' - urban or 'R' - rural)
5. famsize - family size (binary: 'LE3' - less or equal to 3 or 'GT3' - greater than 3)
6. Pstatus - parent's cohabitation status (binary: 'T' - living together or 'A' - apart)
7. Medu - mother's education (numeric: 0 - none, 1 - primary education (4th grade), 2 - 5th to 9th grade, 3 secondary education or 4 higher education)
8. Fedu - father's education (numeric: 0 - none, 1 - primary education (4th grade), 2 -5th to 9th grade, 3 - secondary education or 4 - higher education)
9. Mjob - mother's job (nominal: 'teacher', 'health' care related, civil 'services' (e.g. administrative or police), 'at_home' or 'other')
10. Fjob - father's job (nominal: 'teacher', 'health' care related, civil 'services' (e.g. administrative or police), 'at_home' or 'other')
11. reason - reason to choose this school (nominal: close to 'home', school 'reputation', 'course' preference or 'other')
12. guardian - student's guardian (nominal: 'mother', 'father' or 'other')
13. traveltime - home to school travel time (numeric: 1 - <15 min., 2 - 15 to 30 min., 3 - 30 min. to 1 hour, or 4 - >1 hour)
14. studytime - weekly study time (numeric: 1 - <2 hours, 2 - 2 to 5 hours, 3 - 5 to 10 hours, or 4 - >10 hours)
15. failures - number of past class failures (numeric: n if 1<=n<3, else 4)
16. schoolsup - extra educational support (binary: yes or no)

17. famsup - family educational support (binary: yes or no)
18. paid - extra paid classes within the course subject (Math or Portuguese) (binary: yes or no)
19. activities - extra-curricular activities (binary: yes or no)
20. nursery - attended nursery school (binary: yes or no)
21. higher - wants to take higher education (binary: yes or no)
22. internet - Internet access at home (binary: yes or no)
23. romantic - with a romantic relationship (binary: yes or no)
24. famrel - quality of family relationships (numeric: from 1 - very bad to 5 - excellent)
25. freetime - free time after school (numeric: from 1 - very low to 5 - very high)
26. goout - going out with friends (numeric: from 1 - very low to 5 - very high)
27. Dalc - workday alcohol consumption (numeric: from 1 - very low to 5 - very high)
28. Walc - weekend alcohol consumption (numeric: from 1 - very low to 5 - very high)
29. health - current health status (numeric: from 1 - very bad to 5 - very good)
30. absences - number of school absences (numeric: from 0 to 93)

The following grades are related with the course subject; some you will use as features, and some as output targets, as described below:

31. G1 - first period grade (numeric: from 0 to 20)
32. G2 - second period grade (numeric: from 0 to 20)
33. G3 - final grade (numeric: from 0 to 20, output target)

***For the classification problem,** convert the output variable from $y \in [0,20]$ into a 5-class categorical value (I, II, III, IV, V) based on the rules below [2]:

I	**II**	**III**	**IV**	**V**
(excellent/very good)	(good)	(satisfactory)	(sufficient)	(fail)
16-20	14-15	12-13	10-11	0-9
A	B	C	D	F

For example, if your mission is to predict G3, you need to replace the original G3 column with a new G3 column based on the above mapping, and use the new G3 as your output value. If you are using G1 as a feature, then leave G1 in numeric form, $x_{31} \in [0,20]$.

Required training and test sets

2 datasets are provided and required: one training set (for you to use as you like for training, validation, cross-validation, etc.); one test set to use on your final model, to estimate its performance on unknowns.

Your Missions

You have 3 missions (3 ML problems to solve). This applies to both the regression and the classification problem:

1. Predict first-period academic performance without any prior academic performance data: remove the G2 and G3 columns from the original dataset, then predict G1.

2. Predict final-period academic performance without any prior academic performance data: remove the G1 and G2 columns from the original dataset, then predict G3.
3. Predict final academic performance using all available prior academic performance data: Keep G1 and G2 columns inside the dataset as features, then predict G3.

Required Reference Systems

You must code, evaluate and report the test performance of the following reference systems.

For the reference systems below, remove the 4 categorical non-binary features, and use unnormalized data to train and test.

For Regression

- Trivial System:
  - A system that always outputs the mean output value $\bar{y}$ from the training set.
- Baseline systems (both are required):
  - 1NN (output value is the same as the nearest training-set data point in feature space)
  - Linear Regression (no regularization)

For 5-class classification

- Trivial System:
  - A system that randomly outputs class labels with probability based on class priors (priors calculated from the training set). Run the trivial system at least 10 times and take the average as the final prediction of the trivial system.
- Baseline system:
  - Nearest Means

Required performance measures to report

For Regression (see dataset 1 description for definitions):

- Root Mean Squared Error (RMSE)
- Mean Absolute Error (MAE)
- R-squared (R^2)

For 5-class Classification:

- Accuracy
- Macro f1-score*
- Confusion matrix

*Comment: for multiclass problems, you will have a precision, recall, and f1 score for each of the classes. Macro f1-score is the mean of f1 score over all classes.

Overall Tips (apply to both classification and regression)

(i) Consider converting the data into a more usable form, e.g. non-binary categorical variables are best converted to binary one-hot variables; binary categorial data (e.g., values M or F), can be converted simply to binary numeric (e.g., values 1 or 0).

(ii) For the systems you choose, design, and optimize, you may include the 4 categorical non-binary features, try normalizing, try adding other features (e.g., nonlinear functions or combinations of the original features), reducing the feature set, other data preprocessing techniques, data engineering techniques, etc.

(iii) The number of features in our dataset may lead to overfitting if you want to implement nonlinear transformations. If you want to include nonlinearities, you can apply nonlinearities selectively (only to some features), and/or perform feature selection or dimensionality reduction.

(iv) For this dataset, for some of the missions you may find only small or moderate improvements over the baseline are obtainable. That is the nature of some datasets. You should be able to get at least a small improvement over the baseline, but no need to fret if you can't do better than that for some of the missions.

Estimated difficulty level (1-5, 1 is easy, 5 is difficult)

3 (moderate)

References

[1] Dataset information: https://archive.ics.uci.edu/ml/datasets/Student+Performance

[2] P. Cortez and A. Silva. Using Data Mining to Predict Secondary School Student Performance. In A. Brito and J. Teixeira Eds., Proceedings of 5th FUture BUsiness TEChnology Conference (FUBUTEC 2008) pp. 5-12, Porto, Portugal, April, 2008, EUROSIS, ISBN 978-9077381-39-7.

http://www3.dsi.uminho.pt/pcortez/student.pdf

3. Algerian Forest Fires dataset

Description written by: *Thanos*

Summary description

This dataset contains weather data from 2 regions in Algeria over the period of 3 months and the goal is to predict if a fire occurred at any day within that period. To create a real-world scenario, we want to predict if there will be a fire in a future date as provided by the dataset. The fire prediction is based on weather data collected from the regions.

Problem type: Classification.

Data Description

The dataset contains 12 features, all real valued with no missing values. They are:

1. Date: Date when the data point was collected. **NOTE:** given in day/month/year format
2. Temperature: Max. temperature of the day in degrees Celsius
3. RH: Humidity
4. Ws: Wind speed in km/h
5. Rain: rain level in mm
6. FFMC*: Fine Fuel Moisture Code
7. DMC*: Duff Moisture Code
8. DC*: Drought Code
9. ISI*: Initial Spread Index
10. BUI*: Buildup Index
11. Classes: The label of the dataset (1: there was a fire, 0: not a fire)

Comment: The dataset is quite balanced with 45% of the data points in the "fire" class and 55% in the "not fire" class.

Required data sets: 1 dataset is provided and required. It is required to split the dataset in training and test set. The training set will contain datapoints dated between June and including August from the Date feature. The test set will contain the datapoints of September from the Date feature.

Required performance measures to report

- F1-score
- Accuracy
- Confusion matrix

Required reference systems

You must code, evaluate and report the test performance of the following reference systems.

- **Trivial system:** A system that outputs class assignments (S_1, S_2) at random with probability N_1/N and N_2/N, respectively; N_i is the population of data points with class label S_i, and N is the total population of data points, all based on the *training set*.
- **Baseline system:** Nearest means classifier.

Tips

To improve model performance, and follow the spirit of the problem, you can do the following:

- ***Feature engineering***

  You can create extra features on the whole dataset (training + test) based on the existing/original ones. The new features can reflect statistics of the original ones such as:

  Average/min/max/median temperature/rain/humidity/wind of the last 2/5/7 days

  Such features may be useful to detect patterns that can help us predict fires.

  IMPORTANT: Since we are trying to predict fires in future dates, we must use only **PAST** dates when collecting the statistics. More specifically, if we want to find the max temperature of the past 3 days of Aug. 15, we will assign the max temperature of the dates between 12-14 Aug. for that feature of Aug. 15.

  If we follow the above approach and have for example a new feature of the average rain of the past 7 days, then the test set will have information of the training set. That is, the average rain of the past 7 days of Sept. 4th which belongs to the test set, will have information of Aug. 29th – 31st which belongs to the training set. Thus, you will have to remove the last dates of the training set that would contain information for the test set. (I.e., if you used statistics over the past 7 days, **remove** dates of Aug. 24th to Aug. 31st of the training set for the training process.)

  Similarly, cross validation should account the date feature when splitting into training and validation set. In each cross-validation iteration the training and validation sets should contain data points from continuous dates. For example, in one cv iteration the training set could be from June 1st to July 7th, and from August 1st to August 31st and the validation from July 8th to July 31st. In this case, we would still have to remove the last x days of the training set from June 1st to July 7th and the last x days of the validation set from July 8th to July 31st because the August training set will contain information of the last x days of the validation set. In essence, if a validation set follows a training set we remove the last x days of the training set and if a training set follows a validation set we remove the last x days of the validation set. Thus, sklearn commands for cross validation should be avoided as they randomly split the data. Using such techniques or commands for cross-validation will not result in an accurate model performance and may select a model that will underperform on the test set.

- ***Feature selection or reduction***

  After the feature engineering step, the dataset will have a large number of features, which can slow down model runtime or possibly worsen model performance. It is important to remove some features (or reduce dimensionality of feature space some other way) and keep $D' < D$ number of features that are more useful for the prediction task:

  1) A simple way of doing this is to measure each feature's correlation (such as with Pearson correlation coefficient) with the label and then use the x number of features with the highest correlation.

2) Another way of doing this is using a simple model (such as a linear model) to predict fire using one feature at a time and then use the x number of features with the highest performance.

3) Another useful method (that generally gives better results than 1 and 2 above, but requires more coding and takes more computation time) is Sequential Feature Selection (forward or backward).

4) Other methods for feature selection and dimensionality reduction are discussed in lecture (starting with Lecture 19, p. 9), and Discussion 10, last page.

Estimated difficulty level (1-5, 1 is easy, 5 is difficult):

3-4 (moderate to somewhat challenging)

Sample Code

The following python code shows how to index data points in a dataset between two dates (start_date and end_date). You can use the code to set the training and validation sets to evaluate model performance and to remove the last x days of the training set when required.

```
import datetime # need the datetime library


start_date = '2012-09-01'
end_date = '2012-09-30'
mask_test = (df['Date'] >= start_date) & (df['Date'] <= end_date) #index the dates of September for test set


train_end_date = pd.to_datetime(start_date) - pd.Timedelta(days=6) #subtract 6 days to find end date of training set


mask_train = pd.to_datetime(df['Date']) <= train_end_date #index the training set


X_train = df.loc[mask_train].drop(columns=['Classes'])  #define the train dataframe
y_train = df.loc[mask_train]['Classes']


X_test = df.loc[mask_test].drop(columns=['Classes'])  #define the test dataframe
y_test = df.loc[mask_test]['Classes']
```

References

[1] Dataset information: (https://archive.ics.uci.edu/ml/datasets/Algerian+Forest+Fires+Dataset++#)

[2] Relevant paper: https://link.springer.com/chapter/10.1007/978-3-030-36674-2_37