CS 534 Individual Project Assignment 3 (100 Points)

Due: 11:59 p.m. on 07/27/2025

Project Objective: The goals of this assignment are to help you understand the concepts and the algorithms of Machine Learning that we will discuss/have discussed in Week 5 ~ Week 8.

Note:

*(1) This project is to be done by **EACH STUDENT** individually. No help besides the textbook, materials, and the instructor/TA should be taken. Copying any answers or part of answers from other sources, including your classmates, will earn you a grade of zero.*

*(2) Your program must be developed and implemented in the PyCharm-like IDE, or 10% of the graded score is deducted. Please check and choose the one from here: https://realpython.com/python-ides-code-editors-guide/, as the suggestion. Note that we **DO NOT** use the Jupyter as the IDE.*

(3) Assignments are accepted in their assigned Canvas drop box without penalty if they are received by 11:59PM EST on the due date, or 10% of the graded score is deducted for the late submission per day. Work submitted after one week of its original due date will not be accepted.

Project Deliverables: Submit **ONE zip** file that includes the **MachineFailurePredictorPipeline.py**, **the original raw data ONLY**, and **a PDF file that includes Table 1 and Table 2** indicated below to complete your individual project assignment to Canvas.

In this project, we develop and implement a pipeline that will perform the data pre-preprocessing, train five machine-learning (ML) binary classifiers, select the best-trained models, and then test those best-trained models from which we can select the best one among five of them to develop an application to predict the machine failure in the future.

(1). Study and investigate the raw dataset (**ai4i2020.csv**) provided by http://archive.ics.uci.edu/ml/datasets/AI4I+2020+Predictive+Maintenance+Dataset#.

(2). Read the paper (https://ieeexplore.ieee.org/document/9253083) and its Section II that will help you understand more about the meaning of each attribute in the raw dataset.

(3). In this project, we only consider six input attributes (i.e., Type, Air temperature [K], Process temperature [K], Rotational speed [rpm], Torque [Nm], and Tool wear [min]) and one target variable (i.e., Machine failure).

TWF, HDF, PWF, OSF, and RNF are the failure modes that can be ignored in this project, as they are used to indicate if "Machine failure" should be 1 or 0. That is, if at least one of the above failure modes is true, the process fails and the machine failure label is set to 1, which is the case for 339 datapoints.

**** At the end of the above step, your program code will display all the 10,000 data instances with all the input and output attributes needed to build machine learning models on the console output.**

(4). You also need to perform some data pre-processing that may include categorical data transformation, normalization, and/or standardization.

**** At the end of the above step, your program code will display all the 10,000 data instances with all the transformed input and output attributes needed to build machine learning models on the console output.**

(5). As the raw dataset is severely imbalanced having only 339 datapoints labeled as machine failure ("1"), the rest of the datapoints labeled as machine normal ("0"). For the learning and practice purposes, we will perform the under-sampling by using **RandomUnderSampler**. That is, we would like to use 339 datapoints labeled as machine failure ("1") and 339 datapoints labeled as machine failure ("0") to train and test the ML binary classifiers. In total, the size of our datasets used for this project is 678 datapoints. You can review these two resources, as below, for your reference that will help you complete this step:

(a) The "3. Random under-sampling with imblearn" section from this website (https://www.analyticsvidhya.com/blog/2020/07/10-techniques-to-deal-with-class-imbalance-in-machine-learning/) and

(b) The "RandomUnderSampler" from this website (https://imbalanced-learn.org/stable/references/generated/imblearn.under_sampling.RandomUnderSampler.html?highlight=randomundersampler).

**** At the end of the above step, your program code will display all the 678 data instances with all the transformed input and output attributes needed to build machine learning models on the console output.**

(6). After (5), please divide the above pre-processed dataset into training (80%) and testing (20%) sets (https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html). Using the training portion to do the following step and **DO NOT** touch the testing dataset at this step.

More specifically, you needs to perform the 5-fold cross validation (https://scikit-learn.org/stable/modules/cross_validation.html#computing-cross-validated-metrics) on the training data to develop and implement **EACH** ML model by fine-tuning their hyperparameters until one set of their parameter values of each model delivers the best performance in terms of its MCC-score (https://scikit-learn.org/stable/modules/generated/sklearn.metrics.matthews_corrcoef.html). The ML models with the indicated hyperparameters that you need to develop include:

(a) Multi-layer Neural Network (*hidden_layer_sizes*, *activation*, *learning_rate*): https://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPClassifier.html#sklearn.neural_network.MLPClassifier
(b) Support Vector Machine (*C*, *kernel*, *gamma*): https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html#
(c) K-Nearest Neighbors (*n_neighbors*, *p*, *algorithm*): https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html
(d) Decision Tree (*criterion*, *max_depth*, *ccp_alpha*): https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html#sklearn.tree.DecisionTreeClassifier
(e) Logistic Regression (*penalty*, *C*, *solver*): https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html

In addition to the required parameters above, if there are other parameters (optional) that you would like to fine-tune each ML model further, please feel free to do it and include those parameters and their values in your submission, i.e., **your program code and Table 1 & 2**.

You can decide to use either GridSearch or RandomizedSearch or both to learn the best set of hyperparameter values of each ML model. The following sites will help.

(a) Hyperparameter Tuning: https://scikit-learn.org/stable/modules/grid_search.html#
(b) GridSearch: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html
(c) RandomSearch: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.RandomizedSearchCV.html

**** At the end of the above step, your program code will display the below Table 1 on the console output.**

ML Trained Model	Its Best Set of Parameter Values	Its MCC-score on the 5-fold Cross Validation on Training Data (80%)
Multi-layer Neural Network	(*hidden_layer_sizes*, *activation*, *learning_rate*)	
Support Vector Machine	(*C*, *kernel*, *gamma*)	
K-Nearest Neighbors	(*n_neighbors*, *p*, *algorithm*)	
Decision Tree	(*criterion*, *max_depth*, *ccp_alpha*)	
Logistic Regression	(*penalty*, *C*, *solver*)	

(7). After (6), use the testing dataset to evaluate the performance among all the best-trained models obtained from the above Table 1 and then select the one that beats all the other four models in terms of their MCC-scores.

**** At the end of the above step, your program code will display the below Table 2 on the console output, as well as indicate which the ML model among five should be the chosen one that will be used to predict the machine failure in the future.**

ML Trained Model	Its Best Set of Parameter Values	Its MCC-score on Testing Data (20%)
Multi-layer Neural Network	(*hidden_layer_sizes*, *activation*, *learning_rate*)	
Support Vector Machine	(*C*, *kernel*, *gamma*)	
K-Nearest Neighbors	(*n_neighbors*, *p*, *algorithm*)	
Decision Tree	(*criterion*, *max_depth*, *ccp_alpha*)	
Logistic Regression	(*penalty*, *C*, *solver*)	

(8). Submit the **original raw dataset**, **a PDF file that includes Table 1 and Table 2 above**, and your **pipeline** (**MachineFailurePredictorPipeline.py**), including (3), (4), (5), (6), and (7) using the main() function as the starting point, to Canvas.

That is, when your TA executes your pipeline with your provided raw data, all the above outputs in (3), (4), (5), (6), and (7) will be displayed on your TA's computer console output. You are highly suggested to implement the multiple functions for the above steps.

```
def main():
    pass

if __name__ == "__main__":
    main()
```

Grading Criteria: Your answers must be complete and clear.

Checkpoints	**Points Possible**
a. Deliverables: **MachineFailurePredictorPipeline.py**, **the Original Raw Data**, and **a PDF file that includes Table 1 and Table 2**	5 Points
b. Proper Naming Conventions and Program Documentation on Your Codes	5 Points
c. STEP 3 Process	10 Points
d. STEP 4 Process	15 Points
e. STEP 5 Process	15 Points
f. STEP 6 Process	25 Points
g. STEP 7 Process	15 Points
h. STEP 8 Process	10 Points
Total	100 Points

NOTE: If your selected final model's MCC is the top two in the testing dataset in this assignment of the class, you will add 10 bonus points, as the extra credits, to IP3.