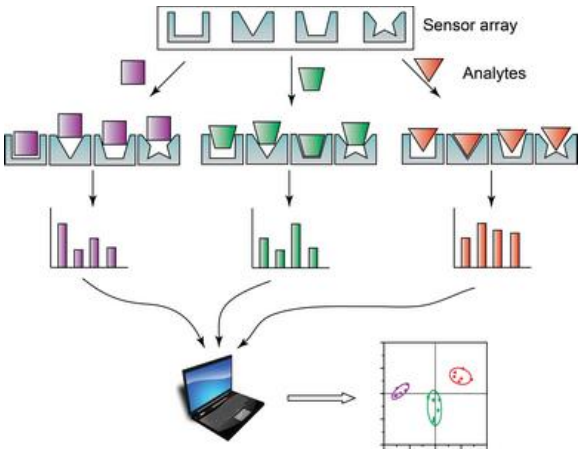
ISAT 341 - Machine Learning and Data Science
Machine Learning Confidential Sensor Data Project
(Teams of two)

** Do not discuss with *anyone* outside of your team. JMU Honor Code violation if you do! **



Objectives

To demonstrate the ability to: Complete an end-to-end data science / machine learning project.

MACHINE LEARNING PROJECT CHECKLIST

Machine Learning projects. Eight main steps (read carefully):

1. Frame the problem and look at the big picture.
2. Get the data.
3. Explore the data to gain insights.
4. Prepare the data to expose the underlying data patterns to Machine Learning algorithms.
5. Explore many different models and short-list the best ones.
6. Fine-tune your models and combine them into a great solution.
7. Present your solution.
8. Launch, monitor, and maintain your system

The Dataset

The data is stored in CSV files: Sensor_Data_Confidential_341Project_DataSetX. (X= 1,2,3,4…)

*****The data source as well as the exact nature of the data is confidential. *****

Each data instance contains 12 real-valued input attributes. Each input attribute represents a sensor designed to detect the presence of one of two groups of substances. As an alternative, the sensor readings may represent a 'false alarm'.

- Substance 1 is represented by the value 'one' in the class attribute column.
- Substance 2 is represented by the value 'two' in the class attribute column.
- A false alarm is represented by the value 'three' in the class attribute column.

There are four csv files with different datasets. You are to perform your analysis on the dataset assigned to your team. There are several combinations of classes in the csv files such as:

- class one: 474 instances
- class two: 399 instances
- class three: 466 instances
- class one: 473 instances
- class two: 400 instances

*****NOTE: When a sensor temporarily goes offline, it records a BAD data value of -9999. *****

Instructions

Data Preprocessing:

1. Download the Excel spreadsheet that contains the data to be used by your team from Canvas (it is in a CSV file format that you are already familiar with).
2. **WARNING: Under NO condition are you to alter any data in the spreadsheet/csv file.**

Machine Learning Pipeline:

You are to build machine learning predictive models from the dataset by using the tools from your earlier jupyter notebooks (python, scikit-learn, pandas, numpy, matplotlib)

3. Performing detail exploratory data pre-processing and data analysis:
    - Use Pandas to load your data into a dataframe
    - Clean data as necessary (this is **new** but real-world data often needs cleaning)
    - Perform relevant statistical analysis
    - Create Data Visualizations
    - Scale the data before training your model(**new**)
4. Build appropriate machine learning models from dataset
    - Create Appropriate Training and Testing Datasets
    - Create Predictive Models
5. Perform Model Evaluation
    - Model Score and Accuracy Model
    - Predict class membership probabilities (and class label names-**new**)
    - Confusion Matrix (this is **new** but an important process after a model is deployed)
    - Other Metrics (two-model statistical comparison)

6. Professionally present your findings to your audience

Implementation Details and Code Snippets

The above six steps are a *minimum and you are expected to follow ALL them*. You may use any of the algorithms (see below), techniques and code that we have developed and studied this semester. In addition, you are also ***REQUIRED*** to research and ***USE EACH*** of the following in your project. Code snippets on *some* **new** procedures are being supplied for you as you peruse this document.

1) Use Pandas dataframe **to find bad or missing data**.
    a. Use replace() and dropna() with NumPy to remove bad or incomplete data from the dataframe during your analysis.

```
df = df.replace(-9999.0, np.nan)
df = df.dropna()
df.head(10)
```

2) Plot bar charts using pandas dataframe (*plot the mean value of the sensors*)

```
#get column names
columns = df.columns.tolist()

#exclude last column (class)
features =len(columns)-1
columns=columns[:features]

#get and print the mean values for all sensors
print('The mean values are: \n{}'.format(df.mean(numeric_only=True)))

#store mean values
mean_values=df[:].mean(numeric_only=True)

#Plot the data with bar charts
plt.figure(figsize=(12, 5))
plt.bar(columns, mean_values )
plt.show()
```

3) Create feature matrix and target vector.
4) Use SciKit-Learn's Label Encoding on your target vector. **Use my variable names.** Perform each task indicated by the comments.

```
# import label encoder
from sklearn.preprocessing import LabelEncoder

# instantiate integer encoder
label_encoder = LabelEncoder()

# encode the class labels
y_encoded =

# print the categorical class labels we encoded (note the underscore!)


# print a few encoded values using python slice


# print the shape of the encoded classes
print('encoded shape:', y_encoded.shape)
```

5) Scale your data (use Standard Scaler)

```
from sklearn.preprocessing import StandardScaler

sc = StandardScaler()
sc.fit(X_train)
X_train_std = sc.transform(X_train)
X_test_std = sc.transform(X_test)
```

6) Train multiple ML models during same session
    a. You **MUST** use the ***three*** *algorithms from previous labs*. One of them **MUST** be logistic regression and the other **MUST** be K-Nearest Neighbor (with K=10, K=50, K-200) when predicting class-membership probabilities.

i. This means the third option is scikit-learn's linear support vector classifier. This model does not have a method to predict class membership probabilities.

b. **You MUST** properly use model evaluation metrics (accuracy, **confusion matrix**, **classification report**, etc.)

c. Your Models **MUST** predict class membership probabilities and associated class label names

7) You can use other techniques and tools such as numpy that may be required to complete the project, but **you cannot consult with any person to complete the project other than your teammate.**

8) You should create a markdown cell in your notebook summarizing and comparing the (i) models you used (ii) the performance metrics (accuracy, etc.) (iii) the classification results from the confusion matrix and classification report, and (iv) the Two-Model Comparison using the Statistical Significance technique given in your notebook.

9) Serializing (saving and re-loading) your Machine Learning Models

a. To receive full credit for this part you must test the saved and re-loaded classifiers on an instance of "unknown" data and show that it correctly classifies the instance.

```
# assign classifier to serialize
saved_classifier = ???
# Serialization with Python's Pickle
import pickle
import os

dest = os.path.join('classifier', 'pkl_objects')
if not os.path.exists(dest):
    os.makedirs(dest)

pickle.dump(saved_classifier, open(os.path.join(dest, 'classifier.pkl'), 'wb'), protocol=4)
```

```
# load the saved the trained classifier into memory
import pickle
import re
import os
os.chdir('classifier')
classifier_reloaded = pickle.load(open(os.path.join('pkl_objects', 'classifier.pkl'), 'rb'))
```

10) **FINALLY**: You should also, **at the END of your notebook** give a *brief* (¼ page) report (in a markdown cell) of your analysis detailing your understanding of the analysis, the machine learning models and their comparative performance, statistical insights, etc. (include reference website links if used).

Grading

Your grade will be determined by:

(i) the execution of all code and markdown in your Jupyter Notebook
(ii) the quality, organization, and output of your code
(iii) how thorough of an analysis you perform and correctness of your answers throughout the analysis.
(iv) the presentation and annotation of your Jupyter Notebook (see previous labs and comments on annotation, image, html, etc.)
(v) your 1/4-page summary report at the end of your notebook

Deliverables

Submit your completed notebook and all associated files, images, etc. to canvas as **a zipped file** *with properly structured subfolders and NO unnecessary files*. You will lose points if you do not adhere to these guidelines.