Homework01: Build ML workflow for Text classification

100 Points Possible

4/28/2025

Attempt 1

In Progress

NEXT UP: Submit Assignment

Add Comment

Unlimited Attempts Allowed

Details

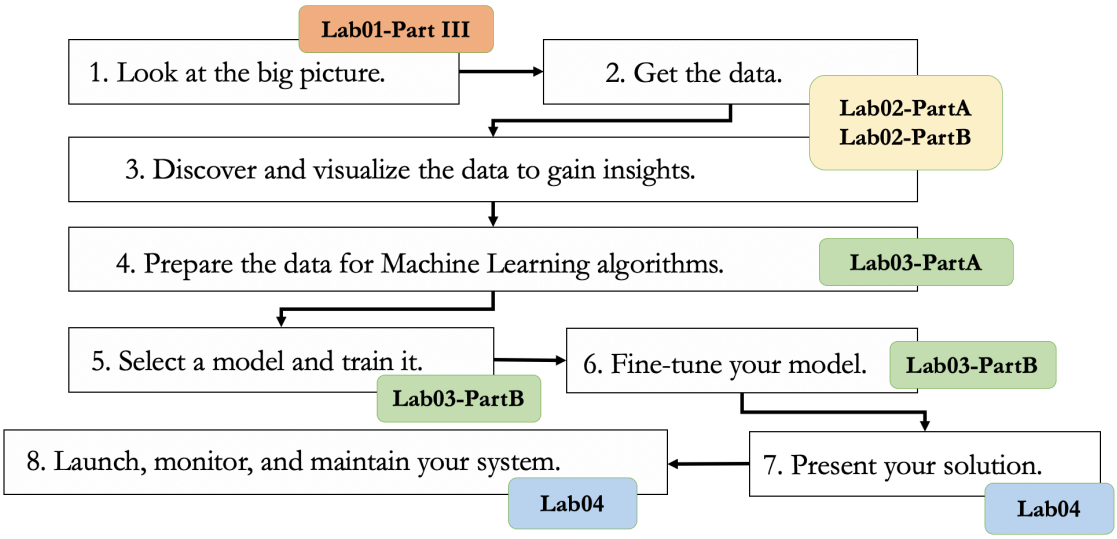
Homework01: Build ML workflow for Text classification

Topic:	**Explore machine leaning methods for Text classification**
TextBook Access	**Tutorial: Access the textbook through the SLU library (https://canvas.slu.edu/courses/68524/pages/tutorial-access-the-textbook-through-the-slu-library)**
Techniques:	Machine learning methods for data analysis using sklearn, numpy, matplotlib, pandas
Collaboration Policy:	**The homework must be completed individually.** **Discussion to resolve programming errors is allowed, but sharing notebooks or code directly is not permitted.** Any academic integrity violation will be reported. Any duplicate solutions from different group's submissions will not be graded. More details regarding academic integrity can be found at **Important Notice: Academic Integrity Policy ($CANVAS_OBJECT_REFERENCE$/discussion_topics/ga7b69e7ca8a9206f25db943959bc349f)**
Submission Deadline: :	11:59pm, Monday, April 28

Note: This homework will serve as reference tutorial for Final Project.

Part I: Develop a Machine Learning workflow for classification

Review End-to-End Data Science Pipeline



Overview

This homework is designed to learn how an ML end-to-end workflow can be generalized to text classification tasks.

Task 1. Before we start the assignments, all students should review the topics of Text Processing Slides **Lecture8_TextProcessing_FeatureExtraction.pdf (https://canvas.slu.edu/courses/68524/files/6037962?wrap=1)**

Module 1. Frame the problem and look at the big picture.

Objective: Build End-to-End Machine Learning Classification Workflow on Sentiment Analysis (Text Classification)

Learning outcomes :

1. This task aims to practice one application in the field of natural language processing (NLP), called the sentiment analysis. This is also the text classification problem, similar to other applications such as spam email filter (i.e., detecting spam email).
2. Train a machine learning model to classify the text/messages into Positive (+1) or Negative (0).
3. To train the model, we need to prepare the raw text to get training dataset.

Application: Machine Learning for Text Preprocessing

❑ **Example: Working with text data (NLP)**

1. Sentiment Analysis

✓ **Explore example here:**

https://huggingface.co/spaces?sort=modified&search=sentiment+analysis

Note: For Text Processing Analysis, please refer to instructions provided in Page: Hw01-Tutorial-TextProcessing (https://canvas.slu.edu/courses/68524/pages/hw01-tutorial-textprocessing)

Module 2. Get the original data

Task: Prepare the dataset for text classification

Follow steps in [Hw01-Tutorial-TextProcessing (https://canvas.slu.edu/courses/68524/pages/hw01-tutorial-textprocessing)](https://canvas.slu.edu/courses/68524/pages/hw01-tutorial-textprocessing) to prepare the yelp comment dataset and load the data on Google Colab

Sample loaded data:

	reviews	class
0	If you haven't gone here GO NOW!	1
1	Try them in the airport to experience some tas...	1
2	The restaurant is very clean and has a family ...	1
3	I personally love the hummus, pita, baklava, f...	1
4	Come hungry, leave happy and stuffed!	1
...	...	...
995	The food was delicious, our bartender was atte...	1

Task Requirement: prepare the training/test data into the following four variables (sentence_train, sentence_test, y_train, y_test) with proper dimensions.

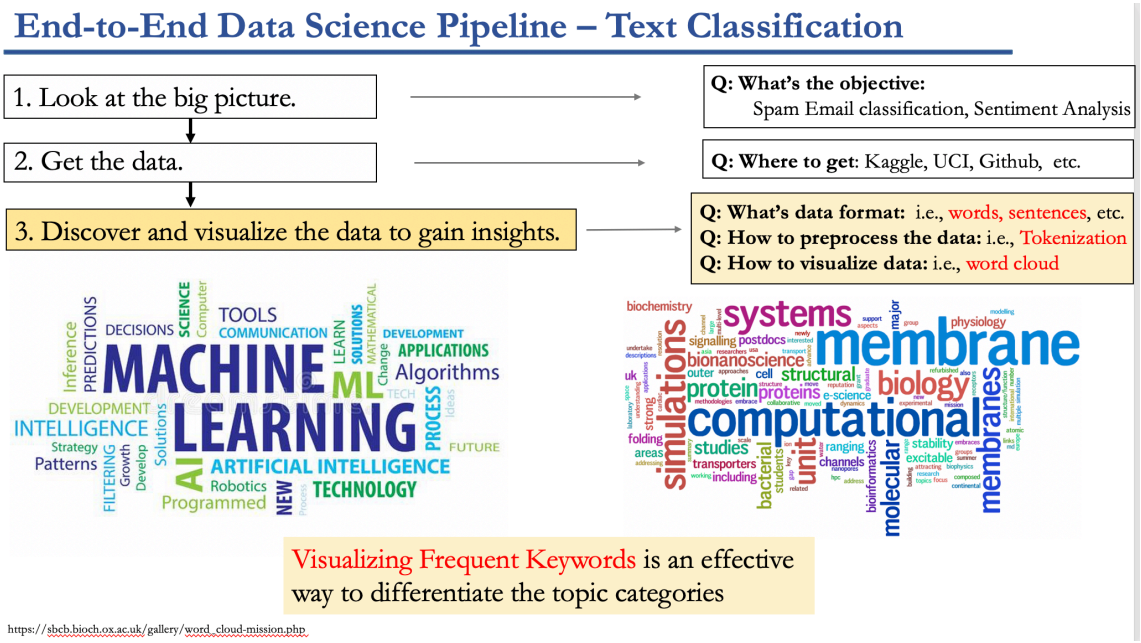
The proper sample printout information should be (the number is not necessary matching the number in image below):

```
sentences_train.shape:  (670,)
sentences_test.shape:  (330,)
y_train.shape:  (670,)
y_test.shape:  (330,)
```

Module 3. Explore and visualize the data to gain insights.

Task M3.1.a. Report the frequency of classes (Positive, Negative classes) in train, and test set. Are they balanced?

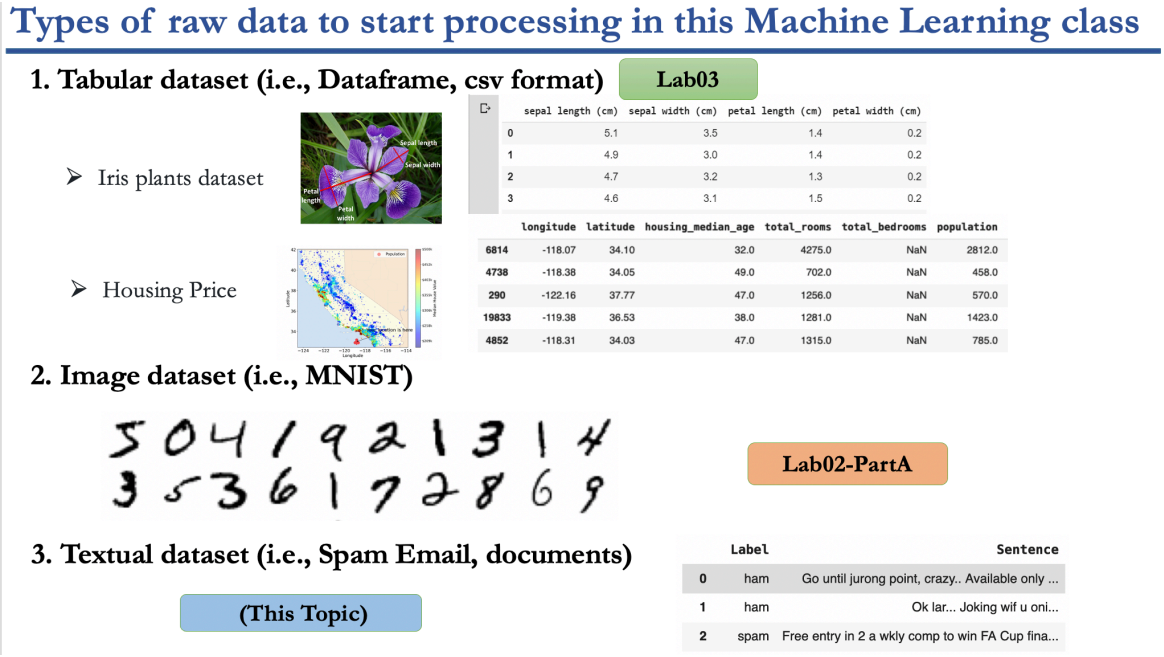
Task M3.1.b. Visualize the Positive texts and Negative texts in the training data using WordCloud



Note: Please refer to demo codes in **https://www.geeksforgeeks.org/generating-word-cloud-python/ (https://www.geeksforgeeks.org/generating-word-cloud-python/)** for wordcloud visualization. Using ChatGPT for visualization assistance is also recommended. Please refer to the rules in **proper_use_of_ChatGPT.pdf (https://canvas.slu.edu/courses/68524/files/6255075?wrap=1) (https://canvas.slu.edu/courses/68524/files/5736289?wrap=1)**).

Module 4. Prepare the training data for Machine Learning algorithms.

Our objective in this step is to convert both image data and text data into a consistent Tabular Format data for classifical Machine Learning algorithm



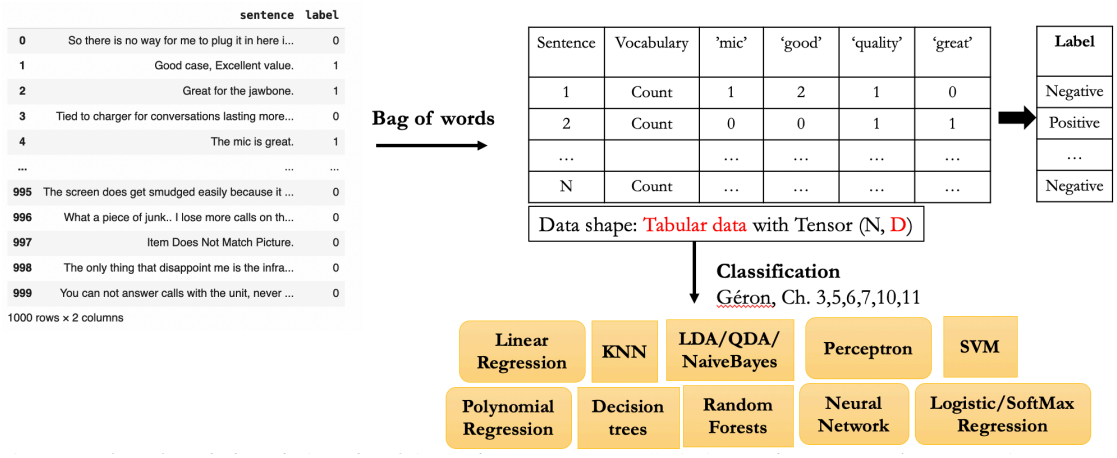
	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)
0	5.1	3.5	1.4	0.2
1	4.9	3.0	1.4	0.2
2	4.7	3.2	1.3	0.2
3	4.6	3.1	1.5	0.2

	longitude	latitude	housing_median_age	total_rooms	total_bedrooms	population
6814	-118.07	34.10	32.0	4275.0	NaN	2812.0
4738	-118.38	34.05	49.0	702.0	NaN	458.0
290	-122.16	37.77	47.0	1256.0	NaN	570.0
19833	-119.38	36.53	38.0	1281.0	NaN	1423.0
4852	-118.31	34.03	47.0	1315.0	NaN	785.0

	Label	Sentence
0	ham	Go until jurong point, crazy.. Available only ...
1	ham	Ok lar... Joking wif u oni...
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...

Summary: Machine Learning for Text Preprocessing

❖Which algorithms to use?



Text Processing: Export the final feature count table as well as the their actual class labels into csv file (Homework01_BoW.csv) using Pandas.

We can directly use this new dataset (numeric count table, and class label) for ML model training in the next step.

Sample format:

Instance	Word 1	Word 2	.....	Word K	Class Label
1					
2					
3					
...					
N					

Module 5. Select and Train different machine learning algorithms.

ML method for Text Processing: Build classification models for multi-class & binary classification.

Note: It is important to review what you have done in Lab03-PartB: End-to-End Machine Learning Project (Model training & Model Selection & Evaluation) (https://canvas.slu.edu/courses/68524/assignments/538875) to complete the following model training & evaluation.

Task M.5.1 Run K-nearest-neighbors algorithm using functions in sklearn (sklearn.neighbors.KNeighborsClassifier):

Note: Before running the function, read the official demonstration of this function in https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html (https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html)

Answer the following questions in the notebook

Task M.5.1.1. According to official document of sklearn.neighbors.KNeighborsClassifier, summarize what hyper-parameters available for hyper-parameter tuning

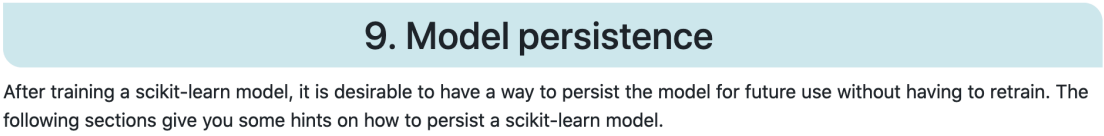
Task M.5.1.2. Training the KNN model (try any K value as you wish) on the training set

```
from sklearn.neighbors import KNeighborsClassifier
knn_model = KNeighborsClassifier(n_neighbors = ?, metric = ?)
knn_model.fit(........)
knn_model.predict(........)
```

Note: Refer to **Lab03-PartB: End-to-End Machine Learning Project (Model training & Model Selection & Evaluation) (https://canvas.slu.edu/courses/68524/assignments/538875)**

Task M.5.1.3. Follow Step 15 of **Lab03-PartB (https://canvas.slu.edu/courses/68524/assignments/538875)**, Save the trained model to disk. Write codes to re-load the model to answer the remaining questions.

Read tutorial in **https://scikit-learn.org/stable/model_persistence.html (https://scikit-learn.org/stable/model_persistence.html)**



9. Model persistence

After training a scikit-learn model, it is desirable to have a way to persist the model for future use without having to retrain. The following sections give you some hints on how to persist a scikit-learn model.

Task M.5.1.4. Define multiple evaluation metrics, calculate the performance on the training and cross-validation scores in terms of accuracy, precision, recall, f1-score and roc score. Refer to practice in **Lecture Note and Practice Question: Classification_EvaluationMetrics (Sep 23) (https://canvas.slu.edu/courses/68524/assignments/538860)**

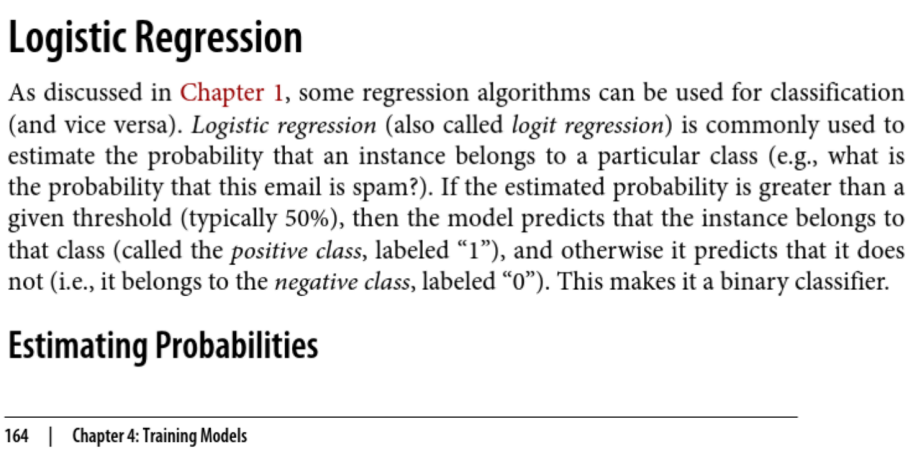
Note 1: You can use the following sklearn functions to get the evaluation metrics:

- accuracy: **https://scikit-learn.org/stable/modules/generated/sklearn.metrics.accuracy_score.html (https://scikit-learn.org/stable/modules/generated/sklearn.metrics.accuracy_score.html)**
- precision: **https://scikit-learn.org/stable/modules/generated/sklearn.metrics.precision_score.html (https://scikit-learn.org/stable/modules/generated/sklearn.metrics.precision_score.html)**
- recall: **https://scikit-learn.org/stable/modules/generated/sklearn.metrics.recall_score.html (https://scikit-learn.org/stable/modules/generated/sklearn.metrics.recall_score.html)**
- f1-score: **https://scikit-learn.org/stable/modules/generated/sklearn.metrics.f1_score.html (https://scikit-learn.org/stable/modules/generated/sklearn.metrics.f1_score.html)**
- roc score: **https://scikit-learn.org/stable/modules/generated/sklearn.metrics.roc_curve.html (https://scikit-learn.org/stable/modules/generated/sklearn.metrics.roc_curve.html)**

If you use sklearn functions, make sure the order of actual labels and predicted labels should be set correctly in this function.

Task M.5.2 : Logistic regression using sklearn: https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html (https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html)

Review chapter 4, P164-P169. Answer the following questions in your notebook



Logistic Regression

As discussed in Chapter 1, some regression algorithms can be used for classification (and vice versa). *Logistic regression* (also called *logit regression*) is commonly used to estimate the probability that an instance belongs to a particular class (e.g., what is the probability that this email is spam?). If the estimated probability is greater than a given threshold (typically 50%), then the model predicts that the instance belongs to that class (called the *positive class*, labeled "1"), and otherwise it predicts that it does not (i.e., it belongs to the *negative class*, labeled "0"). This makes it a binary classifier.

Estimating Probabilities

164 | Chapter 4: Training Models

Task M.5.2.1. According to official document of sklearn.linear_model.LogisticRegression, summarize what hyper-parameters available for hyper-parameter tuning

Task M5.2.2 Review the codes in chapter 4, P146-P147 to train logistic regression models.

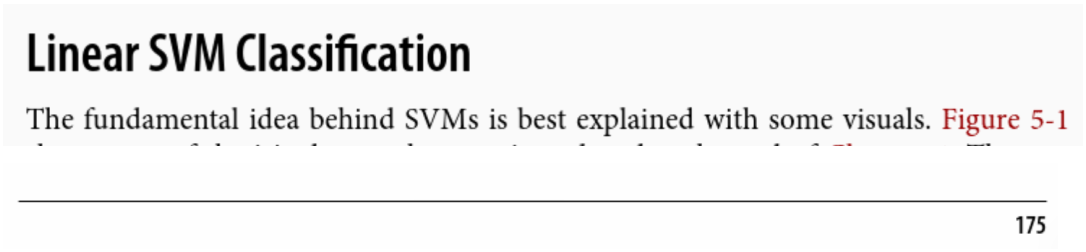
```
from sklearn.linear_model import LogisticRegression
logistic_model = LogisticRegression(penalty = 'l2', C = 1, random_stat
e = 0)
logistic_model.fit(........)
logistic_model.predict(........)
```

Task M5.2.3 Follow **Step 15 of Lab03-PartB (https://canvas.slu.edu/courses/68524/assignments/538875)** , Save the trained model to disk. Write codes to re-load the model to answer the remaining questions.

Task M5.2.4 Define multiple evaluation metrics, calculate the performance on the training and cross-validation scores in terms of accuracy, precision, recall, f1-score and roc score.

Task M.5.3: Support Vector Machine: https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html (https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html)

Read chapter 5, P175-P182. Answer the following questions in your notebook



Linear SVM Classification

The fundamental idea behind SVMs is best explained with some visuals. Figure 5-1

175

Task M.5.3.1. According to official document of sklearn.svm.SVC, summarize what hyper-parameters available for hyper-parameter tuning

Task M.5.3.2. Review the codes in chapter 5, P178-P180 to build several SVM models.

```
from sklearn.svm import SVC
# define linear kernel, P178
svm_model_linear = SVC(C = ? )
svm_model_linear.fit(........)
svm_model_linear.predict(........)

# define polynomial kernel, P179
svm_model_polynomial = SVC(kernel = ?, degree = ?, C = ? )
svm_model_polynomial.fit(........)
svm_model_polynomial.predict(........)


# define Gaussian RBF kernel, P180
svm_model_rbf = SVC(kernel = ?, gamma = ?, C = ? )
svm_model_rbf.fit(........)
svm_model_rbf.predict(........)
```

Task M.5.3.3. Save the trained model of each method to disk. Write codes to re-load the model to answer the remaining questions.

Task M.5.3.4. Define multiple evaluation metrics, calculate the performance on the training and cross-validation scores in terms of accuracy, precision, recall, f1-score and roc score.

Task M.5.4: Random Forest: **https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html (https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html)**

Review chapter 7, P219-P222. Answer the following questions in your notebook

Random Forests

As we have discussed, a random forest[10] is an ensemble of decision trees, generally trained via the bagging method (or sometimes pasting), typically with `max_samples` set to the size of the training set. Instead of building a `BaggingClassifier` and

Task M.5.4.1. According to official document of sklearn.ensemble.RandomForestClassifier, summarize what hyper-parameters available for hyper-parameter tuning

Task M.5.4.2. Review the codes in chapter 7, P220 to build the random forest models.

```
from sklearn.ensemble import RandomForestClassifier
random_forest_model = RandomForestClassifier(n_estimates = ?, max_leaf_nodes
= ?)
random_forest_model.fit(........)
random_forest_model.predict(........)
```

Task M.5.4.3. Save the trained model to disk. Write codes to re-load the model to answer the remaining questions.

Task M.5.4.4. Define multiple evaluation metrics, calculate the performance on the training and cross-validation scores in terms of accuracy, precision, recall, f1-score and roc score.

Module 6. Fine-tune your models.

For Text Processing

Task M.6.1 Perform 10-fold cross-validation and hyper-parameter tuning for all models

You can get the best model from GridSearchCV with cross-validation: **https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html (https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html)**

Follow the sample codes in chapter 2, P91-P94, or **Step 13a-c of Lab03-PartB (https://canvas.slu.edu/courses/68524/assignments/538875)**

Fine-Tune Your Model

Let's assume that you now have a shortlist of promising models. You now need to fine-tune them. Let's look at a few ways you can do that.

Grid Search

(1) K Nearest-neighbors : **https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html (https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html)**

```
hyperparameter_set = {'n_neighbors': [????????]}
knn = KNeighborsClassifier()
# write codes for the parameter tuning
# get the best model from the parameter tuning results
# do your evaluation here......
```

(2) Logistic regression: **https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html (https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html)**

```
hyperparameter_set = {'C': [?????????]}
from sklearn.linear_model import LogisticRegression
logistic_model = LogisticRegression()
# write codes for the parameter tuning
# get the best model from the parameter tuning results
# do your evaluation here......
```

(1) Support Vector Machine : **https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html (https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html)**

```
hyperparameter_set = {'kernel': [????????]}
svc = SVC()
# write codes for the parameter tuning
# get the best model from the parameter tuning results
# do your evaluation here......
```

(2) Random Forest: **https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html (https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html)**

```
hyperparameter_set = {'n_estimates': [?????????]}
rf_model = RandomForestClassifier()
# write codes for the parameter tuning
# get the best model from the parameter tuning results
# do your evaluation here......
```

Task M.6.2 Return the best model from the hyper-parameter tunning step, and Save to the disk

Module 7. Present your evaluation results (i.e., confusion matrix, F1-score)

For Text Processing

Task M.7.1 Based on the best model you get, calculate the performance on the training and test data set in terms of accuracy, precision, recall, f1-score and roc score

Task M.7.2 Summarize all evaluation results on training, cross-validation scores, and test data set from Task M.6 into the Pandas Table for method comparison

An example of a performance comparison is:

	Methods	Accuracy	Precision	Recall	F1-score	AUC score
0	Logistic Regression	0.754422	0.794444	0.860582	0.826192	0.695619
1	K-Nearest Neighbors	0.791837	0.840394	0.855567	0.847913	0.756536
2	Support Vector Machine	0.823810	0.868263	0.872618	0.870435	0.796774
3	Random Forest	0.761905	0.822532	0.827482	0.825000	0.725581

A sample codes to organize results into the above table is:

```
results_data = {'Methods': ['Logistic Regression', 'SVM', 'Random Forest','KNN'],
 'Accuracy': [????, ????,???? ,???? ],
 'Precision':[????, ????,???? ,???? ],
 'Recall':[????, ????,???? ,???? ],
 'F1-score':[????, ????,???? ,???? ],
 'AUC score':[????, ????,???? ,???? ]}


evaluation_metrics_results=pd.DataFrame(results_data)
```

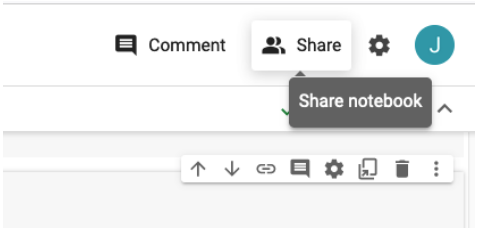
Module 8. Package the software, release, and maintenance.

Requirement: Complete Part II for deploying the models on web application using Gradio. You can either building web application within Google Colab, or huggingface.

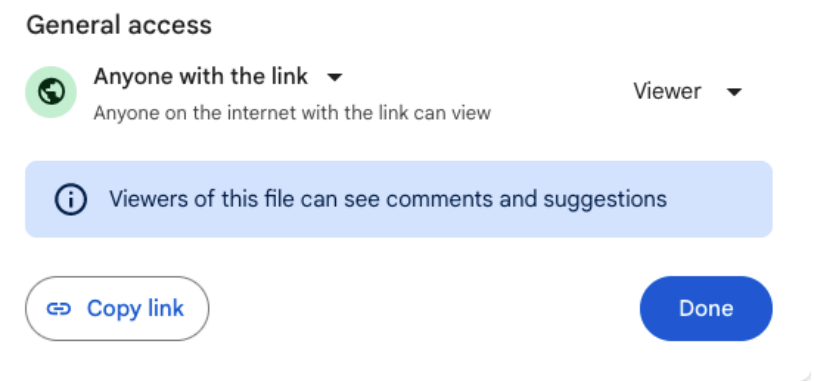
Submission requirement for Part I: Coding Notebook

Please share your google colab link with us to facilitate grading. Really appreciate your efforts. See steps below

Step 1: Click 'Share' on top right corner of Google Colab



Step 2: Set 'General Access' to 'Anyone with the link', and click 'Copy Link'



Step 3: Copy & Paste your Google Colab link in this textbox

Part II: Deploy the machine learning models on Gradio or HuggingFace (Optional for undergraduate student as bonus point)

Following our lab: **Lab04: Build and Host Machine Learning Demos with Gradio (https://canvas.slu.edu/courses/68524/assignments/538862)** ,design web applications for each of the above applications.

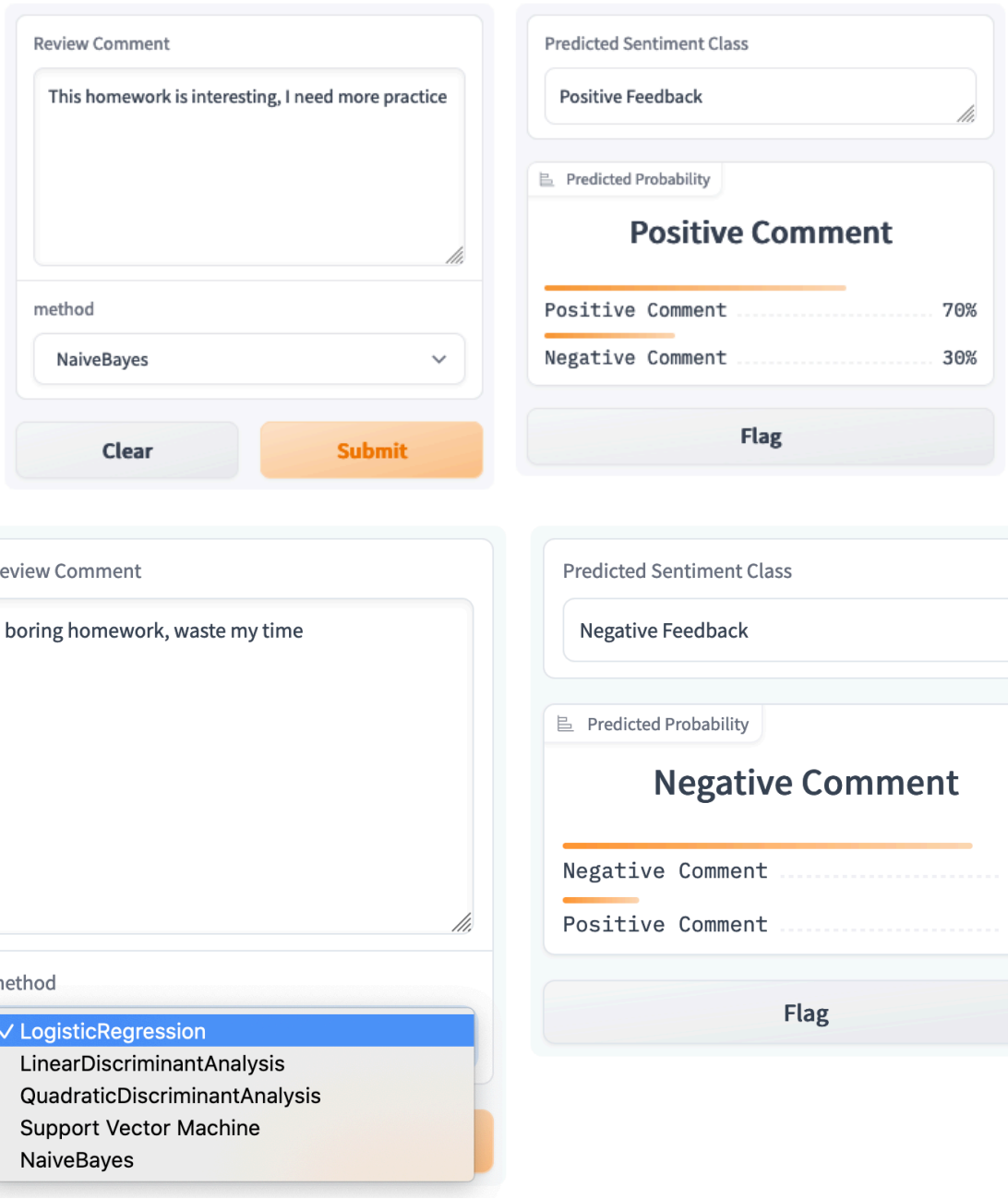
Web application for Text Processing can be designed as any of the following style:

1: input 1: user's review comment

2: input 2: drop down menu for machine learning models (**https://gradio.app/docs/#dropdown (https://gradio.app/docs/#dropdown)**)

3: output: predicted class probabilities of classes from the selected machine learning model

An example implementation is attached below:



Submission requirement for Part II: Web application

- If your web application is implemented inside Google Colab, please share the colab link with us to facilitate grading..
- If your web application is implemented inside huggingFace, please share your huggingface link with us to facilitate grading.

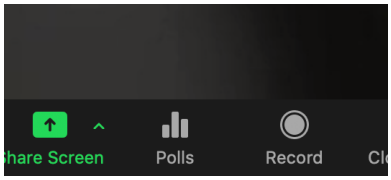
Part IV: Homework Presentation

Submission Requirement:

Every student needs to complete the following parts before the deadline.

(15 Points) **Part IV:** Every student needs to give 10-15 min demo to describe how your group completes all tasks. Every student needs to have your own recording. The detailed presentation activities are attached below:

Step 1. Open zoom (myslu -> zoom), find the following buttons



Step 2. Click "share screen" button

Step 3. Click "record" button

Step 4. Present the following items:

a. (10 points) summarize **how the end-to-end ML workflow is applied to Text Sentiment Analysis problem**

c. (5 points) Summarize what you have learned in this homework.

Limit your presentation no longer than 15min. Please do not read codes during presentation. You should explain the purposes of codes in each step.

Step 5. Stop recording

Step 6. Submit the recording link as following format in the text box of submission page

Zoom recording link:XXXXXXXXX
Passcode: XXXXXX

(do not download recording video)

Choose a submission type

Text | Web URL | Upload | More

Submit Assignment