COS40007 Artificial Intelligence for Engineering

Week 2 Studio Activities

ILO	Understand Feature Engineering and Ground Truth.
Aim	• Learn how to pre-process data • Learn how to apply feature engineering • Understand the data labelling process and develop ground truth • Understand how to develop a Machine Learning model and compare different features
Resources	Books: 1. Prosise, Jeff. Applied machine learning and AI for engineers. "O'Reilly Media, Inc.", 2022. 2. Raschka, Sebastian, Yuxi Hayden Liu, and Vahid Mirjalili. Machine Learning with PyTorch and Scikit-Learn: Develop machine learning and deep learning models with Python. Packt Publishing Ltd, 2022. Web Resources: 1. https://www.geeksforgeeks.org/machine-learning-with-python/
Requirements for submission to be marked as complete	Demonstrate the table of outcomes to your tutor

Data Pre-processing

In Studio-1, you conducted some exploratory data analysis (EDA) on your selected dataset. EDA is the first Data pre-processing in Machine Learning. After performing the EDA, you can remove irrelevant data points. The next step of data pre-processing is Feature Engineering. Also, sometimes, you may need to create a labelled dataset if ground truth information is unavailable. We will learn these two processes in the current studio.

Studio Activity 1: Class labelling / creating ground truth data

In this case, we can use our example dataset of cement manufacturing. The outcome variable of this problem was concrete compressive strength, which was measured in the original dataset as a numerical value in MPa. Let us simplify the problem with the following definition of compressive strength and establish that as ground truth.

1: Very low: if strength is below 20
2: Low: if strength is between 20 and 30
3: Moderate: if strength is between 30 and 40
4: Strong: if strength is between 40 and 50
5: Very strong: if strength is over 50

So, we have five labels in the target variable.

1) Now write a program (using Python pandas) that converts the numerical value strength to a categorial value (between 1 to 5 described above)

Hint: How you can do it using pandas

1) Load the CSV dataset in the pandas' dataframe
2) Iterate each row and read the strength column
3) Create an if then else block to check the strength value per the above definition.
4) Assign the new categorial value (1 to 5) to the strength if one of the conditions is satisfied.
5) Write the converted data frame to a new CSV file ("converted_concrete.csv")

2) Plot the distribution of classes in a bar chart. What did you observe? Is there primarily an equal distribution of 5 classes, or is there an imbalance? (i.e., imbalance means if there are significant differences such as more than 200 between 2 class samples)

Studio Activity 2: Feature Engineering

Now, let us do some feature engineering to simplify our dataset for developing a machine learning model. To do so, let us look at the outcomes of EDA described in section 5.6 here.

1) First, simplify the 'age' feature, as this has an integer value compared to others. How many unique age values are there? Can you convert this to a categorical value (1,2,3,…) for age (1,3,7,…)? The age distribution appears to contain only a few unique values, which will simplify your model computation.
2) Normalise seven features other than age (cement, slag, ash, water, superplastic, coarseagg, finding) using minmaxtsscale and save the file with the eight features (including age feature computed in step 1) as "normalised_concrete.csv".
3) Create four new composite features by taking the covariance of the normalised value of (cement, slag), (cement, ash), (water, fineeg) and (ash + Superplastic) as in points 2-4 of section 5.6, such composite features are recommended from EDA. Name these four features as cement_slag), cement_ash, water_fineeg and (ash_Superplastic).

4) Save these newly generated 12 (1+7+4 in the above step 1-3) features along with class label (as 1 to 5) as “features_concrete.csv”.

Studio Activity 3: Feature selection

The first observation in EDA says, “Except 'Cement', 'Water', 'Superplastic' and 'Age' features, all other features have a very weak relationship with concrete 'Strength' features and do not account for making statistical decisions (of correlation).” So, let us keep these four composite features and drop other features from “features_concrete.csv”. save your new data as “selected_feature_concrete.csv”.

Studio Activity 4: Model development

Develop a decision tree classifier with the following (Note that all cases have a 70-30% split for training and validation data) and compute the accuracy of each model.

1) converted_concrete.csv (all features without normalisation and without composite features)
2) normalised_concrete.csv (all features with normalisation and without composite features)
3) features_concrete.csv (all features with normalisation and containing composite features)
4) selected_feature_concrete.csv (selected features with normalisation)
5) selected_converted_concrete.csv (selected feature without normalisation) [here select only 'Cement', 'Water', 'Superplastic' and 'Age' as a feature]

Here is a sample codebase (for selected_converted_concrete.csv) to develop a decision tree classifier

Load libraries import
pandas as pd
from sklearn.tree import DecisionTreeClassifier # Import Decision Tree Classifier from sklearn.model_selection import train_test_split # Import train_test_split function
from sklearn import metrics #Import scikit-learn metrics module for accuracy calculation

```
col_names = [cement, 'water', 'superplastic', 'age', 'strength'] #
load dataset
concrete = pd.read_csv("normalised_concrete.csv ", header=None, names=col_names)

feature_cols = [cement, 'water', 'superplastic', 'age', 'strength']
X = concrete [feature_cols] # Features y
= concrete.strength # Target variable
```

```
Split the dataset into a training set and test set
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=1) # 70% training and
30% test
# Create Decision Tree classifier object clf
= DecisionTreeClassifier()

# Train Decision Tree Classifier
clf = clf.fit(X_train,y_train)

#Predict the response for test dataset y_pred =
clf.predict(X_test)

# Model Accuracy, how often is the classifier correct?
print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
```

Studio Activity 5: Summarisation

You will obtain five different accuracy values once you develop the decision tree classifier using five different feature sets. In a Word document, summarise our outcome in terms of accuracy using the following table and demonstrate this to your tutor.

Model 1	Model 2	Model 3	Model 4	Model 5
[accuracy in %]	[accuracy in %]	[accuracy in %]	[accuracy in %]	[accuracy in %]

- Which feature set did you obtain the highest accuracy for, and for which did you obtain the lowest?

Once you complete the table and find the answer to the above question, show this to your tutor.

Next Steps:

The assessment Task for Studio 2 can now be attempted and submitted via Canvas.