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 data labelling process and developing 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 outcome to your tutor

Data Pre-processing

In Studio-1 you have conducted some exploratory data analysis (EDA) on your selected dataset. EDA is the first of Data pre-processing in Machine Learning. After performing EDA you can get rid of irrelevant data points. The next step of data pre-processing is Feature Engineering. Also sometime you may need to create a labelled dataset if ground truth information is not available. We will learn these 2 processes in current studio .

Studio Activity 1: Class labelling / creating ground truth data

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

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

So, we have 5 labels in the target variable.

1) Now write a program (using python pandas) that convert the numerical value strength to a categorial value (between 1 to 5 descried above)

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

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

Studio Activity 2: Feature Engineering

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

1) First simplify the 'age' feature as this has integer value than others. How many unique age values are there? Can you covert this to a categorical value (1,2,3,…) for age (1,3,7,…) as it looks like age distribution contains only few unique values. This will simplify your model computation.
2) Normalise 7 features other than age (cement, slag, ash, water, superplastic, coarseagg, fineagg) using minmaxtsscale and save the file with the 8 features (including age feature computed in step 1) as "normalised_concrete.csv"
3) Create 4 new composite features by taking the covariance of normalised value of (cement, slag), (cement , ash), (water, fineeg) and (ash + Superplastic) as in point 2-4 of section 5.6 such composite features are recommended from EDA. Name these 4 features as cement_slag), cement_ash, water_fineeg and (ash_Superplastic)

4) Save these newly generated 12 (1+7+4 in 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 are having very weak relationship with concrete 'Strength' feature and does not account for making statistical decision (of correlation).” So, let us keep these 4 features and 4 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 decisionTree classifier with the following (Note all cases do 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 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 dataset into 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 classifer object
clf = DecisionTreeClassifier()

# Train Decision Tree Classifer
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

Once you develop the decision tree classifier using 5 different feature set you will obtain 5 different accuracy value. Now in a word document create a summary of 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 %]

- For which feature set you obtained the highest accuracy and for which you obtained a the lowest one ?

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

Next Steps:

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