COS40007 – Artificial Intelligence for Engineering

Portfolio Assessment 1

Name: Ngo Nguyen Phuc

Student ID: 1064777599

Contents

Selecting Dataset

The dataset that I chose for Portfolio 1 is Water Quality in Water Engineering

Here is the link to the dataset:

https://www.kaggle.com/datasets/uom190346a/water-quality-and-potability

The reason for choosing this dataset:

Since I study computer science and the most of the datasets offered are for engineering students, I selected the Water Quality dataset because it piqued my curiosity. There are simply ten features in this dataset, which is sufficient to familiarize me with the steps involved in building a machine learning model. Furthermore, given the decline in water quality in my native nation, water pollution is another major worry of mine. In addition to giving me a better understanding of the entire model-building process, experimenting with this dataset has improved my ability to select the best feature to maximize the training process.

Implementation

Exploratory Data Analysis (EDA) process

Some of the most crucial elements of the Water Quality dataset that will determine the outcome of the Feature Engineering stage are as follows, which I have determined after completing the Exploratory Data Analysis (EDA) process at Studio 1:

The dataset is primarily skewed to the right, with the exception of sulfate and chloramines, which are somewhat skewed to the left.

The remaining characteristics do not demonstrate a viable association with the goal variable, which is Potability, with the exception of pH, Solids, Chloramines, and Turbidity. They are therefore unsuitable for decision-making based on correlation.

However, there is a slight positive association between the pH feature and organic carbon. Therefore, the prediction percentage for potability would rise if they were combined as a new attribute (Organic_carbon_ph).

Furthermore, there is a modest link between Solids and Trihalomethanes. Thus, both features might be trihalomethanes_solids, a novel feature.

Lastly, there is a weak positive association between Hardness and Chloramines. Therefore, a new property called chloramines_hardness might be created by combining the two traits.

Class labeling

After looking at the dataset for a while, I think that the goal variable, potability, is binary (0 and 1), which has given the model a clear class to classify, making relabel unnecessary.

In order to better see the distribution of non-portable values (0), portable values (1), and any other values, such as NaN values, I have also made a straightforward bar chart.

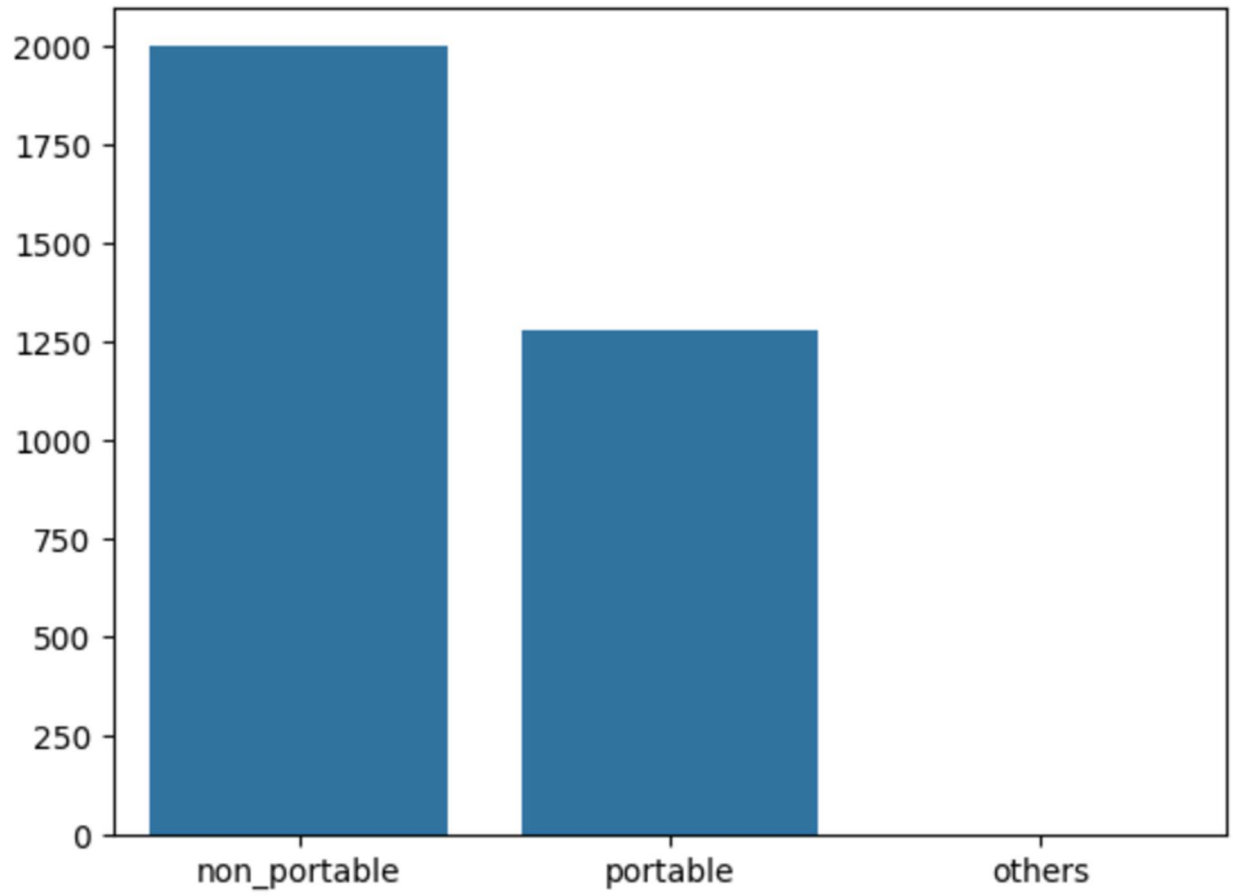


Figure1: Class distribution

All things considered, this bar graph shows an imbalance in the distributions of potability classes:

Non-potable values: The sample size is obviously greater, indicating that over 60% of the data are non-potable.

Less than 40% of the occurrences are drinkable water, as indicated by the reduced sample size compared to non-portable values.

Other values: No samples were discovered in this part because the dataset had been thoroughly cleansed.

Feature Engineering

Because the binary description of the objective feature, potability, is used. Since the present value already indicates that 0 is a non-potable value and 1 is a potable value, there is no need to transform it to categorical values.

Normalization

With the exception of the target feature, normalizing every feature will ensure that every numerical feature has the same scale between 0 and 1, improving the model's overall performance.

```
from sklearn.preprocessing import MinMaxScaler

normalised_df = water_df.copy()
scaler = MinMaxScaler()
normalise_features = [i for i in normalised_df.columns if i != 'Potability']
normalised_df[normalise_features] = scaler.fit_transform(normalised_df[normalise_features])
print("Transformed Data")
normalised_df.head()
```

Figure 2: Using MinMaxScaler for Normalization

Composite features

By combining Organic carbon and pH to create Organic_carbon_ph, Chloramines and Hardness to create Chloramines_hardness, and Trihalomethanes_solids to represent the weak positive correlation between Trihalomethanes and Solids, I have added four new features based on the investigation from the Exploratory Data Analysis (EDA) process. I have integrated it using the built-in cov() function from numpy and the covariance approach.

```
features_df['Organic_carbon_ph'] = np.cov(features_df['Organic_carbon'], features_df['ph'])[0,1]
features_df['chloramines_hardness'] = np.cov(features_df['Chloramines'], features_df['Hardness'])[0,1]
features_df['trihalomethanes_solids'] = np.cov(features_df['Trihalomethanes'], features_df['Solids'])[0,1]
```

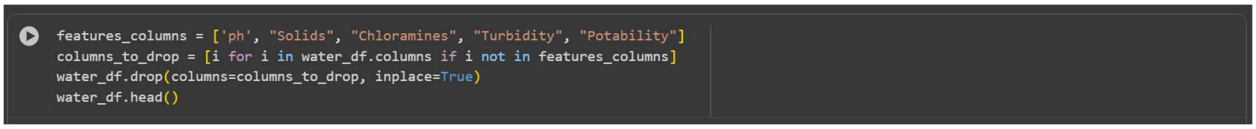
Figure 3: Composite features

Selecting features

According to the EDA method, the only parameters that have a significant impact on potability are pH, solids, chloramines, and turbidity; all other features have very little correlation with potability. As a result, I made the decision to produce both normalized and unnormalized datasets using the chosen features.

```
features_columns = ['ph', "Solids", "Chloramines", "Turbidity", "Potability", "Organic_carbon_ph", "chloramines_hardness", "trihalomethanes_solic
columns_to_drop = [i for i in selected_features_df.columns if i not in features_columns]
selected_features_df.drop(columns=columns_to_drop, inplace=True)
selected_features_df.head()
```

Figure 4: Selecting features for nomalized dataset



```
features_columns = ['ph', "Solids", "Chloramines", "Turbidity", "Potability"]
columns_to_drop = [i for i in water_df.columns if i not in features_columns]
water_df.drop(columns=columns_to_drop, inplace=True)
water_df.head()
```

Figure 5: Selecting features for dataset without nomalization

Developing Model

In order to build our machine learning model to train and test on the original dataset (cleaned_data_water_potability.csv), I created four distinct datasets during the feature engineering and feature selection process that have been both normalized and composited. These datasets include:

- normalized_water_potability.csv dataset includes all features with normalization and without composite features.

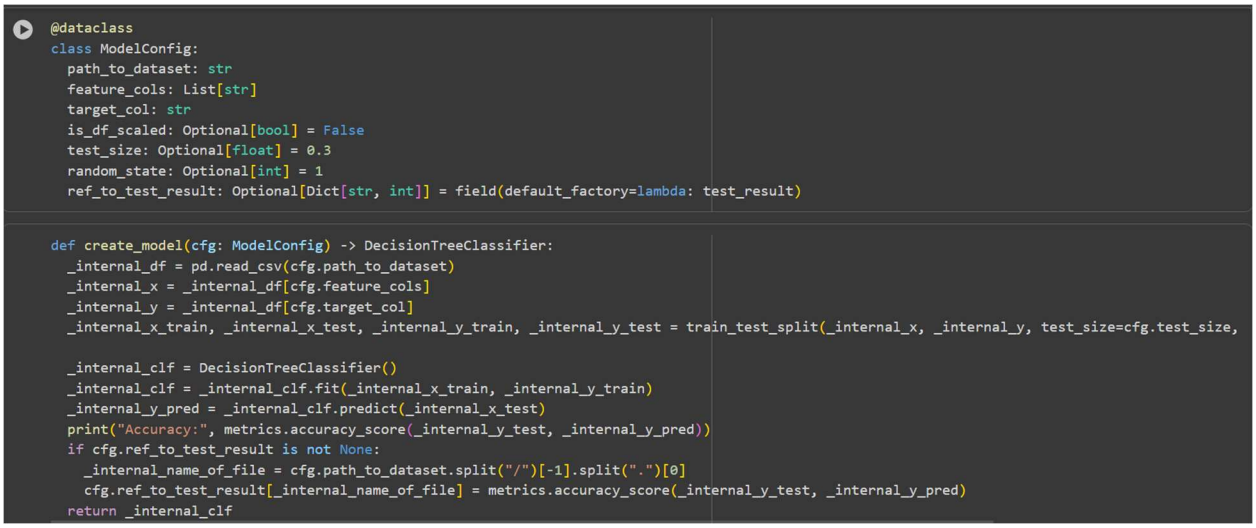
- features_water_potability.csv includes all features with normalization and composite features.

- selected_features_water_potability.csv: This dataset includes features that were chosen during the feature selection procedure, without composite features and normalization.

- selected_features_normalised_water_potability.csv: This dataset includes composite features and selected features that were carried out during the feature selection procedure with normalization.

I have created a machine learning model to train and evaluate the results from the provided datasets using the Decision Tree Classifier from sklearn.

Model:



```
@dataclass
class ModelConfig:
  path_to_dataset: str
  feature_cols: List[str]
  target_col: str
  is_df_scaled: Optional[bool] = False
  test_size: Optional[float] = 0.3
  random_state: Optional[int] = 1
  ref_to_test_result: Optional[Dict[str, int]] = field(default_factory=lambda: test_result)
```

```
def create_model(cfg: ModelConfig) -> DecisionTreeClassifier:
  _internal_df = pd.read_csv(cfg.path_to_dataset)
  _internal_x = _internal_df[cfg.feature_cols]
  _internal_y = _internal_df[cfg.target_col]
  _internal_x_train, _internal_x_test, _internal_y_train, _internal_y_test = train_test_split(_internal_x, _internal_y, test_size=cfg.test_size,

  _internal_clf = DecisionTreeClassifier()
  _internal_clf = _internal_clf.fit(_internal_x_train, _internal_y_train)
  _internal_y_pred = _internal_clf.predict(_internal_x_test)
  print("Accuracy:", metrics.accuracy_score(_internal_y_test, _internal_y_pred))
  if cfg.ref_to_test_result is not None:
    _internal_name_of_file = cfg.path_to_dataset.split("/")[-1].split(".")[0]
    cfg.ref_to_test_result[_internal_name_of_file] = metrics.accuracy_score(_internal_y_test, _internal_y_pred)
  return _internal_clf
```

Figure 6: Developing model

Summary

Result

For testing purposes, the table shows the output of five distinct models using five datasets, four of which are the ones mentioned above and one of which is unnormalized and lacks feature selection or composition.

Model	Accuracy score
cleaned_data_water_potability	55.137 %
normalised_water_potability	55.443 %
features_water_potability	56.256 %
selected_features_normalised_water_potability	53.103 %
selected_features_water_potability	53.510 %

Observation

Based on the provided accuracy score, I have made the following observations:

Maximum precision: features_water_potability.csv are the dataset that obtained the highest accuracy which contained composite features, received scores of 56.256 %. Consequently, it has demonstrated the minor influence of constructing composite features in this dataset.

The feature selection datasets selected_features_water_potability.csv and selected_features_normalised_water_potability.csv on the other hand, have somewhat lower accuracy than the others, achieving 53.510 %and 53.103 %, respectively. Thus, this outcome demonstrates that feature selection is not required for this particular dataset.

Appendix

Source code for Portfolio 1

https://drive.google.com/drive/folders/1WFvRM5ytbHNtIEFw1o3D7P73wMXDaJpm?usp=sharing