0907726 Applied Machine Learning (Fall 2023)

Midterm Exam

	رقم التسجيل:	KEY	الاسم:

==========

Instructions: Time **120** min. Open book and notes exam. No internet usage. Please answer all problems in the respective shaded rectangular spaces. There are two problems. Notice that this exam has 2 CSV files that you need to copy to the working directory of your Python project.

==========

P1. The following Python code loads a dataset that has the features `x1`, `x2`, `x3`, and `x4` and the response `y`. It also creates linear and SVM regressors. Complete this code to achieve the following 6 requirements.

[20 marks]

```
import numpy as np
import pandas as pd
from sklearn.linear_model import LinearRegression
from sklearn.svm import SVR

f = pd.read_csv('dataset1.csv')

lr = LinearRegression()
svr = SVR(kernel="poly", degree=2)
```

1. Find the number of missing values for each feature and the value counts of the categorical feature.

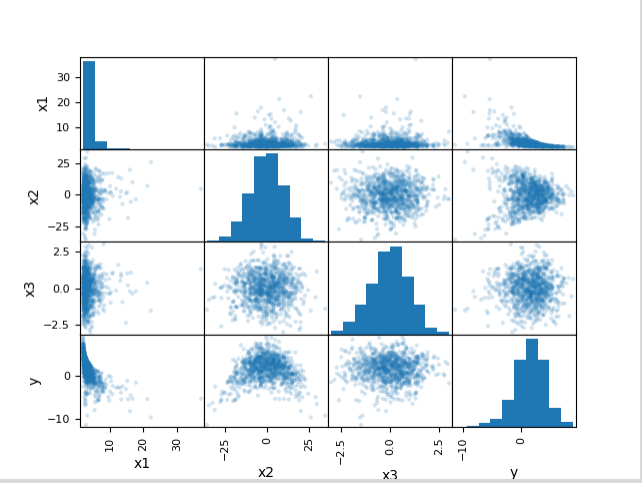
Code	`print(f.info())`
	`print(f.describe())`
	`print(f.x4.value_counts())`

Numbers of missing values	`x1: 10`
	`x2: 4`
	`x3: 13`
	`x4: 0`

Value counts	`White 277`
	`Green 272`
	`Black 226`
	`Red 225`

2. Draw the scatter matrix of this dataset and find the correlation between the response `y` and the numerical features.

Code	`import matplotlib.pyplot as plt`
	`pd.plotting.scatter_matrix(f, alpha=0.2)`
	`plt.show()`
	`print(f.corr().y)`

Scatter matrix	
Correlation	x1 -0.579874 x2 0.039128 x3 0.009580 y 1.000000

3. Drop the feature that has the smallest correlation with the response `y` and split the dataset to 80% train set and 20% test set.

Code

```
f = f.drop('x3', axis=1)
from sklearn.model_selection import train_test_split
train_set, test_set = train_test_split(f, test_size=0.2,
random_state=42)
```

4. Prepare the features to take care of missing values, long tails, normalization, and categorical values.

Code

```
from sklearn.impute import SimpleImputer
imp = SimpleImputer(strategy='median')
X_num = train_set[['x1', 'x2']]
X_num = imp.fit_transform(X_num)

X_num[:, 0] = np.log(X_num[:, 0])

from sklearn.preprocessing import StandardScaler
std_scalar = StandardScaler()
X_num = std_scalar.fit_transform(X_num)

from sklearn.preprocessing import OneHotEncoder
enc = OneHotEncoder()
x4 = enc.fit_transform(train_set[['x4']]).toarray()
```

5. Train the linear and SVR regressors on the prepared train set.

Code

```
X_train = np.c_[X_num, x4]
y_train = train_set['y'].copy()

lr.fit(X_train, y_train)
svr.fit(X_train, y_train)
```

6. Evaluate the two regressors on the test set and report the two RMSEs. Remember to make the necessary preparations for the test set.

Code

```
X_num = test_set[['x1', 'x2']]
X_num = imp.transform(X_num)
X_num[:, 0] = np.log(X_num[:, 0])
X_num = std_scalar.transform(X_num)
x4 = enc.fit_transform(test_set[['x4']]).toarray()
X_test = np.c_[X_num, x4]
y_test = test_set['y'].copy()

from sklearn.metrics import mean_squared_error
yp = lr.predict(X_test)
lr_rmse = mean_squared_error(y_test, yp, squared=False)
print('LR RMSE = ', lr_rmse)

yp = svr.predict(X_test)
svr_rmse = mean_squared_error(y_test, yp, squared=False)
print('SVR RMSE = ', svr_rmse)
```

RMSE

```
LR RMSE =  2.1104137383338246
SVR RMSE =  0.5575093550292866
```

P2. The following Python code loads a dataset that has the features `x1`, `x2`, `x3`, and `x4` and class `y`. It also creates a random forest classifier. Complete this code to achieve the following 2 requirements.

[10 marks]

```
import pandas as pd
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import cross_val_predict
from sklearn.metrics import accuracy_score, confusion_matrix

df = pd.read_csv('dataset2.csv')

forest_clf = RandomForestClassifier(random_state=42)
```

1. Examine this dataset to determine whether any data preparation is needed.

Code	`print(df.info())`
	`print(df.nunique())`
	`print(df['y'].value_counts())`
	
	`import matplotlib.pyplot as plt`
	`df.hist(bins=50, figsize=(12, 8))`
	`plt.show()`

Preparation needed (Yes/No)	`No`

2. Using cross validation of 3 folds, evaluate the accuracy and confusion matrix of the random forest classifier on this dataset.

Code	`X = df.drop('y', axis=1)`
	`y = df['y'].copy()`
	`y_pred = cross_val_predict(forest_clf, X, y, cv=3)`
	
	`# Calculate accuracy`
	`accuracy = accuracy_score(y, y_pred)`
	`print("Accuracy:", accuracy)`
	
	`# Calculate and print the confusion matrix`
	`cm = confusion_matrix(y, y_pred)`
	`print("Confusion Matrix:")`
	`print(cm)`

Accuracy	`Accuracy: 0.786`

Confusion matrix	`Confusion Matrix:`
	`[[211  35   3   1]`
	`[ 24 185  38   3]`
	`[  1  41 177  31]`
	`[  0   0  37 213]]`

<Good Luck>