0907779 Machine Learning (Fall 2018)
Midterm Exam

رقم التسجيل: .... الاسم: ....

==

Instructions: Time **60** min. Open book and notes exam. No electronics. Please answer all problems in the space provided and limit your answer to the space provided. No questions are allowed. There are six problems.

==

P1. Briefly give an answer to each of the following three questions.

[6 points]

1) In supervised learning, what is the main difference between classification and regression?

   Classification is for predicting type and regression is for predicting value

2) What is the main purpose of clustering?

   To determine the intrinsic grouping in a set of unlabeled data.

3) Is the support vector machine classifier instance based or model based?

   Model based.

P2. Write a python code that prints the squares of all even numbers from 0 to 20, i.e., it should print 0, 4, 16, 36, …, 400.

[4 points]

```
# your code goes here

for x in range(21):
    if (x % 2) == 0:
        print(x * x)
```

P3. Write a Python function that prints the sum of an arbitrary number of arguments, e.g., calling it with `my_sum(1, 2.5)` should print `3.5` and calling it with `my_sum(1, 2.5, 3)` should print `6.5`. Moreover, this function should accept an optional argument `no_frac` that truncates the fractional part of the sum, e.g., calling it with `my_sum(1, 2.5, 3.6, no_frac=True)` should print `7.0`.

[5 points]

```
# your code goes here

def my_sum(*args, no_frac=False):
    sum = 0
    for arg in args:
        sum = sum + arg
    if no_frac:
        print(sum // 1)
    else:
        print(sum)
```

P4. Given the information in the box blow and the shown Python code, what is the result of each of the following print statements?

[6 points]

```
>>> data.info()
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20000 entries, 0 to 19999
Data columns (total 3 columns):
x1         20000 non-null float64
x2         20000 non-null float64
y          20000 non-null float64
dtypes: float64(3)
memory usage: 1.1+MB
```

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler

train_set, test_set = train_test_split(data, test_size=0.2)

X_train = train_set.drop("y", axis=1)
y_train = train_set["y"].copy()

scaler = StandardScaler()
scaler.fit(X_train)
X_train_scaled = scaler.transform(X_train)
```

Print Statement	Result
`print(len(train_set))`	1600
`print(X_train_scaled["x1"].mean())`	0.0
`print(X_train_scaled["x2"].std())`	1.0

P5. Given the information in the box blow, complete the following Python code to train an SVM Regressor using the two features to predict the response `y` using the scaled train set and save the trained model to a file named `svm_model.pkl`.

[4 points]

```
>>> data.info()
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20000 entries, 0 to 19999
Data columns (total 3 columns):
x1         20000 non-null float64
x2         20000 non-null float64
y          20000 non-null float64
dtypes: float64(3)
memory usage: 1.1+MB
```

```
from sklearn.preprocessing import StandardScaler
from sklearn.svm import SVR
from sklearn.externals import joblib

X_train = data.drop("y", axis=1)
y_train = data["y"].copy()

scaler = StandardScaler()
scaler.fit(X_train)
X_train_scaled = scaler.transform(X_train)

# your code goes here

svm_reg = SVR()
svm_reg.fit(X_train_scaled, y_train)

joblib.dump(svm_reg, "svm_model.pkl")
```

P6. Complete the following Python code to train a random forest classifier on the training set, predict the classes of the test set, and print the precision and recall scores.

[5 points]

```
from sklearn.datasets import fetch_mldata
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import precision_score, recall_score

mnist = fetch_mldata('MNIST original')
X, y = mnist["data"], mnist["target"]
X_train, X_test = X[:60000], X[60000:]
y_train, y_test = X[:60000], X[60000:]

# your code goes here

forest_clf = RandomForestClassifier (random_state=42)
forest_clf.fit(X_train, y_train)

y_pred = forest_clf.predict(X_test)

print(precision_score(y_train, y_pred))
print(recall_score(y_train, y_pred))
```

<Good Luck>