CS412 - Machine Learning - 2022

Homework 1

100 pts

Goal

The goal of this homework is three-fold:

- Introduction to the machine learning experimental set up
- Gain experience with the Sklearn library
- Learn which evaluation metric to choose in different scenarios

Dataset

JOB-A-THON - March 2022 Dataset contains the customer demographics and past activity with the bank. There is also the target label representing whether the customer will churn (stop working with the bank) or not.

Dataset is taken from: https://www.kaggle.com/datasets/gauravduttakiit/jobathon-march-2022

Download the data from SuCourse

Task

Build a kNN and decision tree classifiers with the scikit library function calls to predict customer churn. Is_Churn is the target variable while the others will be used as features ('Age', 'Gender', 'Income', 'Balance', 'Vintage', 'Transaction_Status', 'Product_Holdings', 'Credit_Card', 'Credit_Category').

Software: You may find the necessary function references here:

http://scikit-learn.org/stable/supervised_learning.html

Submission:

Name this notebook as: **YourName_Surname_hw1.ipynb** from the top left corner (e.g. for Özgür Can Seçkin -> Ozgur_Can_Seckin_hw1.ipynb) Next, download this notebook as a *.ipynb document and upload* it on SuCourse. Also, click on the **"Share"** button on the top right hand side of the page and swicth from **"restricted" to "Anyone with the Link"** under the **"Get Link"** title. Then, **copy the link and paste it on the assignment's comment.**

If necessary, see the first recitation under Week 4 to get a tutorial on how to share your Colab link and how to download the .ipynb file.

Provide the Colab Link Here:

https://drive.google.com/file/d/142s1p3sqvl3MzogIW-oRGLvTIxnKFOxl/view?usp=sharing

##**1) Initialize**

- First make a copy of the notebook given to you as a starter.
- Make sure you choose Connect form upper right.

##**2) Load training dataset** *(5 pts)*

Read the csv file:

```
# Load the Pandas libraries with alias 'pd'
import pandas as pd


# Read data
df = pd.read_csv('/content/train_hw1.csv')
```

##**3) Understanding the dataset** *(5 pts)*

There are alot of functions that can be used to know more about this dataset

- What is the shape of the training set (num of samples X number of attributes) ***(shape function can be used)***
- Get a quick summary of the data **(info can be used)**
- Display the first 5 rows from training dataset ***(head or sample functions can be used)***

Note: Understanding the features, possibly removing some features etc. is an important part in building an ML system.

```
# print shape
print('Data Dimensionality: ')
print(df.shape)

print('\n')

# print the summary for each column
print('Data Summary: ')
df.info()

print('\n')

# print first 5 rows in your dataset
```

```
print('Head of Data: ')
df.head(5)
```

```
Data Dimensionality:
(6650, 11)


Data Summary:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6650 entries, 0 to 6649
Data columns (total 11 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   ID                  6650 non-null   object
 1   Age                 6650 non-null   int64
 2   Gender              6650 non-null   object
 3   Income              6650 non-null   object
 4   Balance             6650 non-null   float64
 5   Vintage             6650 non-null   int64
 6   Transaction_Status  6650 non-null   int64
 7   Product_Holdings    6650 non-null   object
 8   Credit_Card         6650 non-null   int64
 9   Credit_Category     6650 non-null   object
 10  Is_Churn            6650 non-null   int64
dtypes: float64(1), int64(5), object(5)
memory usage: 571.6+ KB


Head of Data:

         ID  Age  Gender         Income     Balance  Vintage  \
0  84e2fcc9   36  Female       5L - 10L   563266.44        4
1  57fea15e   53  Female   Less than 5L   875572.11        2
2  8df34ef3   35  Female  More than 15L   701607.06        2
3  c5c0788b   43  Female  More than 15L  1393922.16        0
4  951d69c4   39  Female  More than 15L   893146.23        1

   Transaction_Status Product_Holdings  Credit_Card Credit_Category
Is_Churn
0                   0                1            0         Average
1
1                   1                1            1            Poor
0
2                   1                2            0            Poor
0
3                   1                2            1            Poor
1
4                   1                1            1            Good
1
```

##**4) Seperate X and y Values** *(5 pts)* First, you have to seperate X (features or independent variables) from y (target variable or dependent variables).

Note that our **dependent variable** is *'Is_Churn'* while the **independent variables** are: *'Age', 'Gender', 'Income', 'Balance', 'Vintage', 'Transaction_Status', 'Product_Holdings', 'Credit_Card', 'Credit_Category'*

Hint: You can use drop method when defining X

```
# Drop ID column:
df = df.drop(columns=['ID'])

# Define X: Features/Independent Variables
X = df[['Age', 'Gender', 'Income', 'Balance', 'Vintage',
'Transaction_Status', 'Product_Holdings', 'Credit_Card',
'Credit_Category']]

# Define y: Target Variable/Dependent Variable
y = df['Is_Churn']
```

##**5) Train - Test Split** *(5 pts)* Since you will be going to apply cross validation, you do not need to use a validation set for now.

```
from sklearn.model_selection import train_test_split

# Split 80-20
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42)
```

##**6) Transform Features Into Numerical Format** *(10 pts)* You can take:

- Gender information as a binary variable
- Income and Credit_Category as ordinal variables

The dictionaries that you are going to use are given below:

```
gender_map = {'Female':1, 'Male': 0}

Income_map = {'Less than 5L': 1,
              '5L - 10L': 2,
              '10L - 15L': 3,
              'More than 15L': 4}

Credit_Category_map = {'Poor': 1,
                       'Average': 2,
                       'Good': 3}

Product_Holdings_map = {'3+': 3}

# You can use replace method to replace the values on training and
test sets:
```

```
# 'Gender' --> gender_map
X_train["Gender"] = X_train["Gender"].replace(gender_map)
X_test["Gender"] = X_test["Gender"].replace(gender_map)

# 'Income' --> Income_map
X_train["Income"] = X_train["Income"].replace(Income_map)
X_test["Income"] = X_test["Income"].replace(Income_map)

# 'Credit_Category' --> Credit_Category_map
X_train["Credit_Category"] =
X_train["Credit_Category"].replace(Credit_Category_map)
X_test["Credit_Category"] =
X_test["Credit_Category"].replace(Credit_Category_map)

# 'Product_Holdings' --> Product_Holdings_map
X_train["Product_Holdings"] =
X_train["Product_Holdings"].replace(Product_Holdings_map)
X_test["Product_Holdings"] =
X_test["Product_Holdings"].replace(Product_Holdings_map)

X_train.head(10)
```

	Age	Gender	Income	Balance	Vintage	Transaction_Status	\
3759	31	1	2	810212.13	5	0	
2728	37	1	1	480472.92	2	1	
1632	30	0	2	700588.71	1	0	
555	66	0	4	559805.67	3	1	
4577	41	0	3	856706.13	3	1	
3126	37	0	2	1047398.94	0	1	
5465	42	1	1	1421413.47	1	0	
4726	43	1	4	946563.84	3	0	
1272	42	1	3	879507.27	0	1	
6224	34	0	3	647110.89	2	1	

	Product_Holdings	Credit_Card	Credit_Category
3759	1	1	1
2728	2	0	2
1632	2	1	1
555	2	1	3
4577	2	1	1
3126	2	1	3
5465	2	1	1
4726	1	0	1
1272	1	1	2
6224	2	1	3

##**7) Select the Best Evaluation Metric** *(25 pts)* Since our bank is very risk averse, it tries to minimize the number of credits that we call "good" but actually are "bad".

		True Label	
		Good	**Bad**
Predicted Label	**Good**	True Positive	False Positive
	Bad	False Negative	True Negative

Which of the following performance metric should we employ in order to select the best model? (You can use the table above) Please write down the correct answer below (under **YOUR ANSWER HERE**) and explain your reasoning in one or two sentences.

1. Accuracy
2. Precision
3. Recall

YOUR ANSWER HERE:

We choose to use "Precision" performance metric because Precision is more focused in the positive class than in the negative class. Therefore, since we want to minimize the False Positives, it is better to use Precision performance metric.

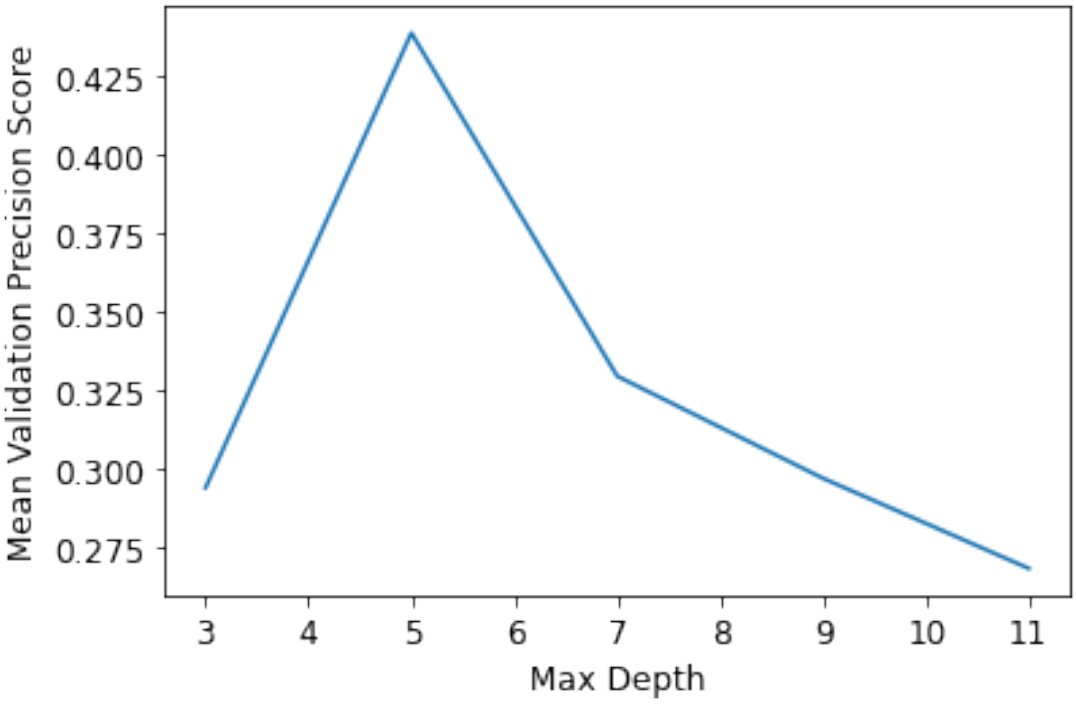
##8) Train a decision tree classifier on train data and do model selection using the validation data *(30 pts)*

- Set number of folds in cross validation as 5 *(5 pts)*
- Set scoring parameter to what you have determined in the last question as `scoring = '...'` *(5 pts)*
- Use grid_search to apply hyperparameter tuning on decision tree classifier with max_depth = 3, 5, 7, 9, 11 *(10 pts)*
- Plot the validation set performance metrics for these settings where x axis denotes the max_depth values and validation performance metrics are on y axis. *(10 pts)*

```
from sklearn.model_selection import GridSearchCV
from sklearn.tree import DecisionTreeClassifier

# Train decision tree classifiers
param_grid = {'max_depth': [3, 5, 7, 9, 11]}

clf = DecisionTreeClassifier()
clf.fit(X_train, y_train)


grid = GridSearchCV(clf,
                    param_grid,
                    scoring='precision',
                    cv= 5,
```

```
                    verbose=1,
                    refit=True)

grid_search = grid.fit(X_train, y_train)

print(grid_search.best_score_)
print(grid_search.best_estimator_)

'''
#Confusion Matrix
from sklearn.metrics import plot_confusion_matrix
plot_confusion_matrix(clf, X_test, y_test)
plt.show()
'''


# Plot errors
import matplotlib.pyplot as plt

plt.plot(param_grid['max_depth'],
         grid_search.cv_results_['mean_test_score'])

plt.xticks(fontsize=12)
plt.yticks(fontsize=12)

plt.xlabel('Max Depth', fontsize=12)
plt.ylabel('Mean Validation Precision Score', fontsize=12)

plt.show()
```

```
Fitting 5 folds for each of 5 candidates, totalling 25 fits

/usr/local/lib/python3.7/dist-packages/sklearn/metrics/
_classification.py:1318: UndefinedMetricWarning: Precision is ill-
defined and being set to 0.0 due to no predicted samples. Use
`zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification
.py:1318: UndefinedMetricWarning: Precision is ill-defined and being
set to 0.0 due to no predicted samples. Use `zero_division` parameter
to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))

0.43892324898267177
DecisionTreeClassifier(max_depth=5)
```



##**9) Evaluate the Best Classifier on Test Set** *(15 pts)*

- Predict the labels of testing data. **Note that grid search cv function will return you the best estimator, so you can use directly the fitted function to make your predictions** and report the accuracy.

```
# test prediction using a decision tree with all default parameters
and ..... min-split value
clf_best = DecisionTreeClassifier(max_depth=5, random_state=42)
clf_best.fit(X_train, y_train)

y_pred = clf_best.predict(X_test)

# Report your accuracy
from sklearn.metrics import precision_score # you can find your
scoring metric in this library

prec = precision_score(y_test, y_pred)
print("Accuracy score of the best model is:", prec)
```

```
Accuracy score of the best model is: 0.42
```

##**10) Submission**

Name this notebook as: **YourName_Surname_hw1.ipynb** from the top left corner (e.g. for Özgür Can Seçkin -> Ozgur_Can_Seckin_hw1.ipynb) Next, download this notebook as a *.ipynb document and upload* it on SuCourse. Also, click on the **"Share"** button on the top right hand side of the page and swicth from **"restricted" to "Anyone with the Link"** under the **"Get Link"** title. Then, **copy the link and paste it on the assignment's comment.**

If necessary, see the first recitation under Week 4 to get a tutorial on how to share your Colab link and how to download the .ipynb file.

Questions?

You can and should ask all your Google Colab related questions under Forums and feel free to answer/share your answer regarding Colab.

You can also ask/answer about which functions to use and what libraries...

However you should **not ask** about the core parts, that is what is validation/test, which one shd. have higher performance, what are your scores etc.