COMP 478

CSUCI - Spring 2022

Lab7: A mini-project

Course Evaluation Result

- Thank You for your feedback!
- Additional resources
- Going through the lab's solution together in the class
- Not having HW and Lab in the same week
- More explanations on the lab assignments
- More examples
- Review session before the midterm

2 bonus points + midterm exam

Announcements

- The solution to Lab 5 is posted.
- The sample questions for the midterm exam are posted. I’ll go through the solutions in our review session (Wed, March 23).
- Midterm Exam:
  - Lecture 1 – Lecture 13 (from the beginning to the end of the LR)

Let’s do a mini project together!!

- Let’s work on the “Wine recognition dataset” from sklearn!

A Real-world ML Problem

- Stroke Prediction Dataset:
  - Binary classification task: +/-
  - columns:

1) id: unique identifier
2) gender: "Male", "Female" or "Other"
3) age: age of the patient
4) hypertension: 0 if the patient doesn't have hypertension, 1 if the patient has hypertension
5) heart_disease: 0 if the patient doesn't have any heart diseases, 1 if the patient has a heart disease
6) ever_married: "No" or "Yes"
7) work_type: "children", "Govt_jov", "Never_worked", "Private" or "Self-employed"
8) Residence_type: "Rural" or "Urban"
9) avg_glucose_level: average glucose level in blood
10) bmi: body mass index
11) smoking_status: "formerly smoked", "never smoked", "smokes" or "Unknown"*
12) stroke: 1 if the patient had a stroke or 0 if not

Dataset source: https://www.kaggle.com/fedesoriano/stroke-prediction-dataset

Data Preprocessing

- Load data:

```
import pandas as pd

df = pd.read_csv('./healthcare-dataset-stroke-data.csv')
```

healthcare-dataset-stroke-data

id	gender	age	hypertension	heart_disease	ever_married	work_type	Residence_type	avg_glucose_level	bmi	smoking_status	stroke
9046	Male	67	0	1	Yes	Private	Urban	228.69	36.6	formerly smoked	1
51676	Female	61	0	0	Yes	Self-employed	Rural	202.21	N/A	never smoked	1
31112	Male	80	0	1	Yes	Private	Rural	105.92	32.5	never smoked	1
60182	Female	49	0	0	Yes	Private	Urban	171.23	34.4	smokes	1
1665	Female	79	1	0	Yes	Self-employed	Rural	174.12	24	never smoked	1
56669	Male	81	0	0	Yes	Private	Urban	186.21	29	formerly smoked	1
53882	Male	74	1	1	Yes	Private	Rural	70.09	27.4	never smoked	1
10434	Female	69	0	0	No	Private	Urban	94.39	22.8	never smoked	1
27419	Female	59	0	0	Yes	Private	Rural	76.15	N/A	Unknown	1
60491	Female	78	0	0	Yes	Private	Urban	58.57	24.2	Unknown	1
12109	Female	81	1	0	Yes	Private	Rural	80.43	29.7	never smoked	1
12095	Female	61	0	1	Yes	Govt_job	Rural	120.46	36.8	smokes	1
12175	Female	54	0	0	Yes	Private	Urban	104.51	27.3	smokes	1
8213	Male	78	0	1	Yes	Private	Urban	219.84	N/A	Unknown	1
5317	Female	79	0	1	Yes	Private	Urban	214.09	28.2	never smoked	1
58202	Female	50	1	0	Yes	Self-employed	Rural	167.41	30.9	never smoked	1
56112	Male	64	0	1	Yes	Private	Urban	191.61	37.5	smokes	1

- Input features and labels:

```
x =  df [['gender','age', 'hypertension','heart_disease', 'ever_married',
          'work_type',  'Residence_type', 'avg_glucose_level',  'bmi',  'smoking_status' , 'stroke']]
```

```
print (x['age'].mean(),x['stroke'].value_counts())
```

Data Preprocessing

- Missing values:
  1. Remove datapoints with missing values
  2. Recover the value:
     - Continuous features: Average/mean, median, …
     - Categorical features: Most frequent category
     - Predict missing values using classification, regression, …
     - ….

```
df = df.dropna()
```

Data Preprocessing

- Handling non-numerical features:

```
x = pd.get_dummies(x)
print (x.columns)
```

```
Index(['age', 'hypertension', 'heart_disease', 'avg_glucose_level', 'bmi',
       'stroke', 'gender_Female', 'gender_Male', 'gender_Other',
       'ever_married_No', 'ever_married_Yes', 'work_type_Govt_job',
       'work_type_Never_worked', 'work_type_Private',
       'work_type_Self-employed', 'work_type_children', 'Residence_type_Rural',
       'Residence_type_Urban', 'smoking_status_Unknown',
       'smoking_status_formerly smoked', 'smoking_status_never smoked',
       'smoking_status_smokes'],
      dtype='object')
```

```
x = x.drop(columns='ever_married_Yes')
print (x.columns)
```

```
Index(['age', 'hypertension', 'heart_disease', 'avg_glucose_level', 'bmi',
       'stroke', 'gender_Female', 'gender_Male', 'gender_Other',
       'ever_married_No', 'work_type_Govt_job', 'work_type_Never_worked',
       'work_type_Private', 'work_type_Self-employed', 'work_type_children',
       'Residence_type_Rural', 'Residence_type_Urban',
       'smoking_status_Unknown', 'smoking_status_formerly smoked',
       'smoking_status_never smoked', 'smoking_status_smokes'],
      dtype='object')
```

Data Preprocessing

- Balance data: (Class +: 209, Class -: 4700)

How to balance data?

1. Down-sample the majority class

```
from sklearn.utils import resample

x_minority = x[x.stroke == 1]
x_majority = x[x.stroke == 0]

x_majority_downsampled = resample(x_majority, replace = False, n_samples = len(x_minority), random_state = 0)
```

2. Up-sample the minority class

```
from sklearn.utils import resample

x_minority = x[x.stroke == 1]
x_majority = x[x.stroke == 0]

x_minority_upsampled = resample(x_minority, replace = True, n_samples = len(x_majority)- len(x_minority), random_state = 0)
```

Training Phase

- Split data into train (70%) & test (30%):

```
Y = x_new['stroke']
X = x_new.drop(columns='stroke')


from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=0.3, random_state=0)
```

- KNN:
  - The best model?
  - The best result?

```
from sklearn.neighbors import KNeighborsClassifier
neigh = KNeighborsClassifier(n_neighbors=1)
neigh.fit(X_train, y_train)
pred = neigh.predict(X_test)

from sklearn.metrics import classification_report

print (classification_report(y_test, pred))
```

How to Tune Threshold?

Predict → predicted class label

Predict_proba → probability associated to each class

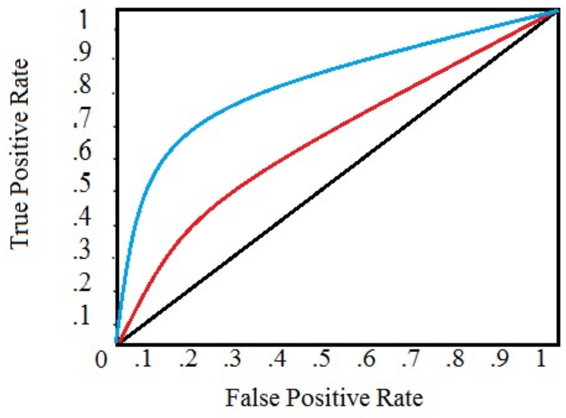
	pred
D1	1
D2	1
D3	0
D4	1
D5	0
D6	0
D7	0

GT label
1
1
1
1
1
0
0

	Class 0	Class 1
D1	0.09	0.91
D2	0.36	0.64
D3	0.55	0.45
D4	0.3	0.7
D5	0.59	0.41
D6	0.95	0.05
D7	0.87	0.13

ROC Curve

- Report the performance of a model:
  - Precision, Recall, f1score, accuracy, confusion matrix, …
  - Receiver Operating Characteristic Curve (ROC curve):
    - Binary classification tasks
    - A graphical plot (TPR vs FPR) for different threshold
    - TPR = $\frac{TP}{TP+FN}$, FPR = $\frac{FP}{FP+TN}$



How to compare different models?

AUC (Area under the ROC Curve)

Image credit: https://www.statisticshowto.com/receiver-operating-characteristic-roc-curve/