

بسم الله الرحمن الرحيم



دانشکده مهندسی برق و کامپیوتر

یادگیری ماشین - تمرین اول

سید مهدی رضوی

استاد : آقای دکتر توسلی پور - آقای دکتر ابوالقاسمی

اسفند ماه ۱۴۰۲

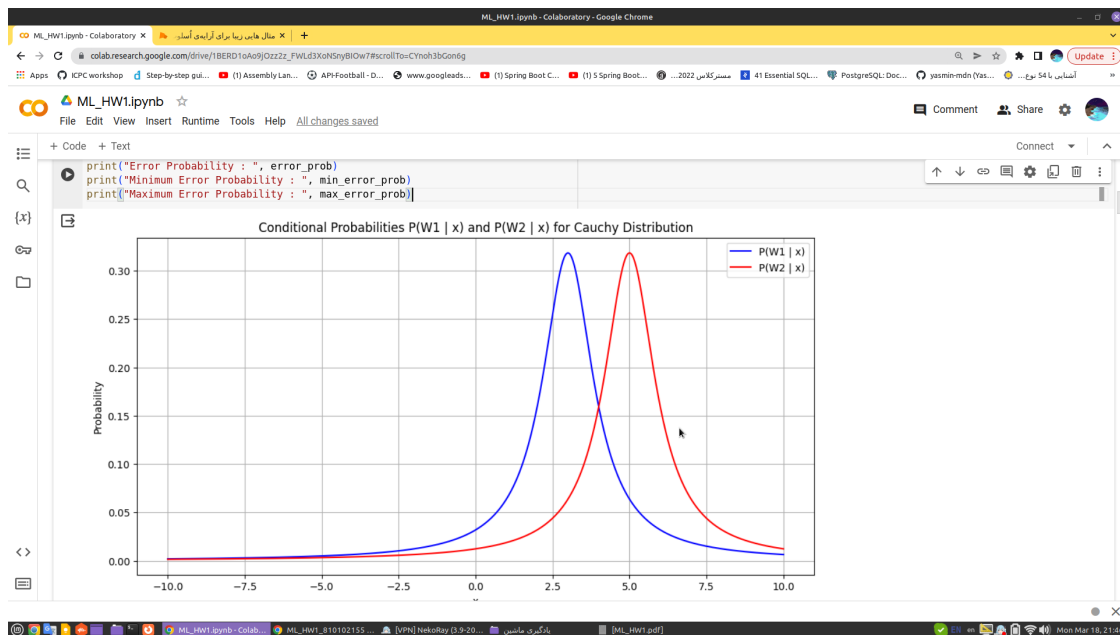
فهرست مطالب

۳	۱ تمرین اول
۵	۲ تمرین دوم
۵	۳ تمرین سوم
۵	۴ تمرین چهارم
۶	۵ تمرین پنجم
۹	۶ تمرین ششم
۱۰	۷ تمرین هفتم
۱۳	۸ تمرین هشتم

فهرست تصاویر

۳	۱ رسم توزیع کوشی برای دو کلاس با مقداردهی های پیرامترها
۴	۲ مرز تصمیم برای دو کلاس
۴	۳ مرز تصمیم برای دو کلاس با توجه به کمینه کردن ماتریس ریسک
۶	۴ محاسبه میانگین و کوواریانس ها
۶	۵ محاسبه میانگین و کوواریانس ها
۷	۶ محاسبه شیب و عرض از مبدأ خط مرز تصمیم
۸	۷ رسم خط مرز تصمیم و ماتریس ریسک
۸	۸ رسم خط مرز تصمیم با استفاده از ماتریس ریسک و احتمال پیشین
۱۰	۹ پیاده سازی بدون کتابخانه از نایوبیز برای مجموعه داده سرطان ریه
۱۱	۱۰ پیاده سازی SKLEARN از نایوبیز برای مجموعه داده سرطان ریه
۱۲	۱۱ پیاده سازی بدون کتابخانه از نایوبیز برای مجموعه داده وب
۱۲	۱۲ پیاده سازی SKLEARN از نایوبیز برای مجموعه داده وب
۱۳	۱۳ نتایج طبقه بندی بر اساس میانگین پیکسل ها
۱۴	۱۴ نتایج طبقه بندی بر اساس واریانس پیکسل ها
۱۵	۱۵ نتایج طبقه بندی بر اساس مد پیکسل ها

تمرین اول



شکل ۱: رسم توزیع کوشی برای دو کلاس با مقداردهی های پارامترها

به منظور رسم شکل فوق می بایستی در ابتدا به هایپرپارامترها مقداردهی اولیه کنیم و سپس از تابع احتمال شرطی توزیع کوشی برای رسم استفاده کنیم.

سپس در شکل ۲ همانطور که مشاهده می فرمایید :

مرز تصمیم را در میانگین بین میانگین های دو توزیع در نظر خواهیم گرفت.

سپس با محاسبه انتگرال ناحیه خطا خواهیم داشت که مساحت این ناحیه برابر با 1.41780 می باشد.

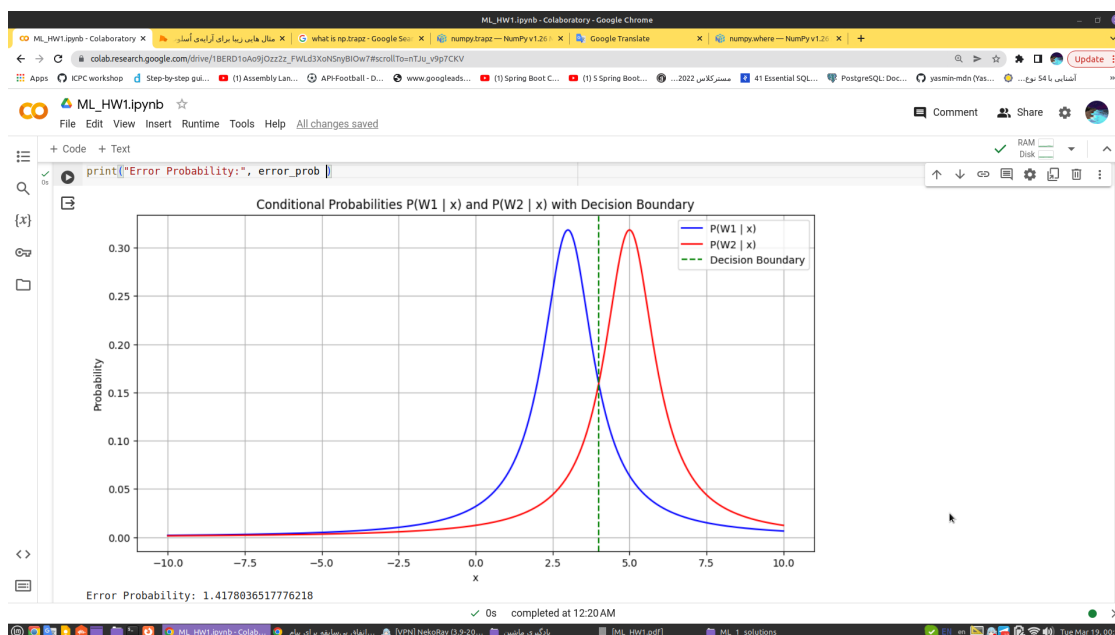
سپس در شکل ۳ :

در صورتی که از ماتریس ریسک استفاده کنیم متوجه خواهیم شد که مرز تصمیم به راست شیفت پیدا می کند ، چرا که جریمه به ازای حدس اشتباه کلاس ۱ دو برابر جریمه به ازای حدس اشتباه کلاس ۲ است.

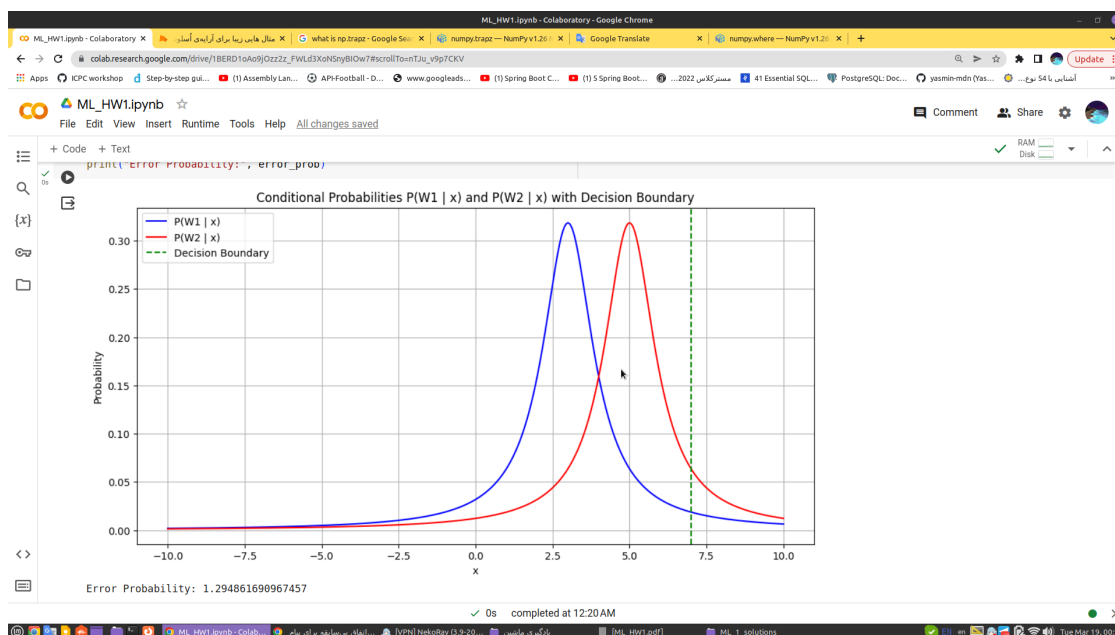
و classifier سعی می کند که ناحیه خطا را کمینه تر کند.

از شکل ۳ نیز پیدا است که این انتگرال ناحیه خطا برابر با 1.29486 خواهد بود .

سایر سوالات تشریحی در فایل Razavi ML HW1 810102155 پاسخ داده شده اند.



شکل ۲: مرز تصمیم برای دو کلاس



شکل ۳: مرز تصمیم برای دو کلاس با توجه به کمینه کردن ماتریس ریسک

۲ تمرین دوم

پاسخ مربوط به این بخش در فایل Razavi ML HW1 810102155 نوشته شده است.

۳ تمرین سوم

پاسخ مربوط به این بخش در فایل Razavi ML HW1 810102155 نوشته شده است.

۴ تمرین چهارم

پاسخ مربوط به این بخش در فایل Razavi ML HW1 810102155 نوشته شده است.

۵ تمرین پنجم

با توجه به نقاط حاضر در صفحه و با استفاده از کتابخانه جبرخطی پایتون موفق به محاسبه میانگین و کواریانس نقاط حاضر در دو کلاس شدیم.

```
print("(W1) Variance of x:", var_x1)
print("(W1) Variance of y:", var_y1)
print("(W1) Covariance between x and y:", cov_xy1)

(W1) Variance of x: 0.5555555555555556
(W1) Variance of y: 0.9876543209876544
(W1) Covariance between x and y: 0.1851851851851852

W2

import numpy as np

# Given data points
x2 = np.array([0.5, -0.5, 1.5, -2, -2, 0.5, 1.5, 1, -1, -2])
y2 = np.array([1, 1, 1, 0, 1, -0.5, -0.5, -1, -1, -1])

# Calculate means
mean_x2 = np.mean(x2)
mean_y2 = np.mean(y2)

# Calculate variances
var_x2 = np.mean((x2 - mean_x2)**2)
var_y2 = np.mean((y2 - mean_y2)**2)

# Calculate covariance
cov_xy2 = np.mean((x2 - mean_x2) * (y2 - mean_y2))

print("(W2) Variance of x:", var_x2)
print("(W2) Variance of y:", var_y2)
print("(W2) Covariance between x and y:", cov_xy2)

(W2) Variance of x: 1.8625
(W2) Variance of y: 0.75
(W2) Covariance between x and y: 0.85
```

شکل ۴: محاسبه میانگین و کواریانس‌ها

```
cov_class2 = np.cov(x2, y2)

# Define Gaussian distributions for each class
dist_class1 = stats.multivariate_normal(mean=mean_class1, cov=cov_class1)
dist_class2 = stats.multivariate_normal(mean=mean_class2, cov=cov_class2)

# Calculate decision boundary
boundary_slope = (mean_class2[1] - mean_class1[1]) / (mean_class2[0] - mean_class1[0])
boundary_intercept = (mean_class1[1] + mean_class2[1] - boundary_slope * (mean_class1[0] + mean_class2[0])) / 2

print("Decision Boundary Equation: y = {:.4f}x + {:.4f}".format(boundary_slope, boundary_intercept))

# Calculate probability of error
prob_error = (1/np.pi) * np.arctan(boundary_slope) + (np.linalg.norm(mean_class1 - mean_class2)**2) / (4 * np.linalg.norm(mean_class1 - mean_class2))
print("Probability of Error:", prob_error)

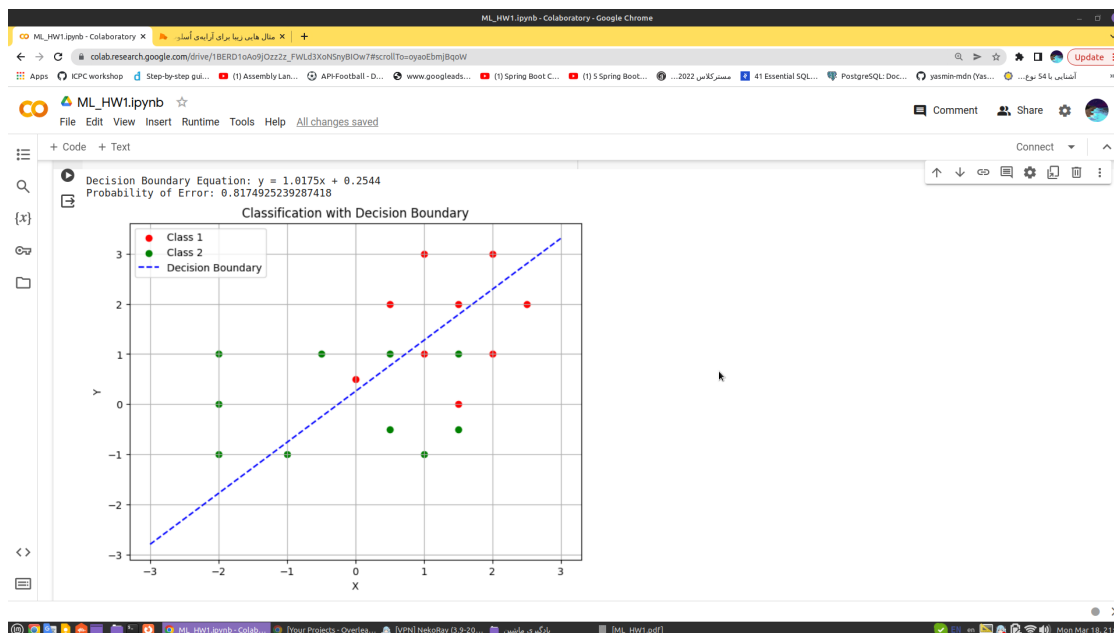
Decision Boundary Equation: y = 1.0175x + 0.2544
Probability of Error: 0.8174925239287418

import numpy as np
import scipy.stats as stats
import matplotlib.pyplot as plt

# Given data points for class 1
x1 = np.array([0, 1.5, 1, 2, 0.5, 1.5, 2.5, 1, 2])
```

شکل ۵: محاسبه میانگین و کواریانس‌ها

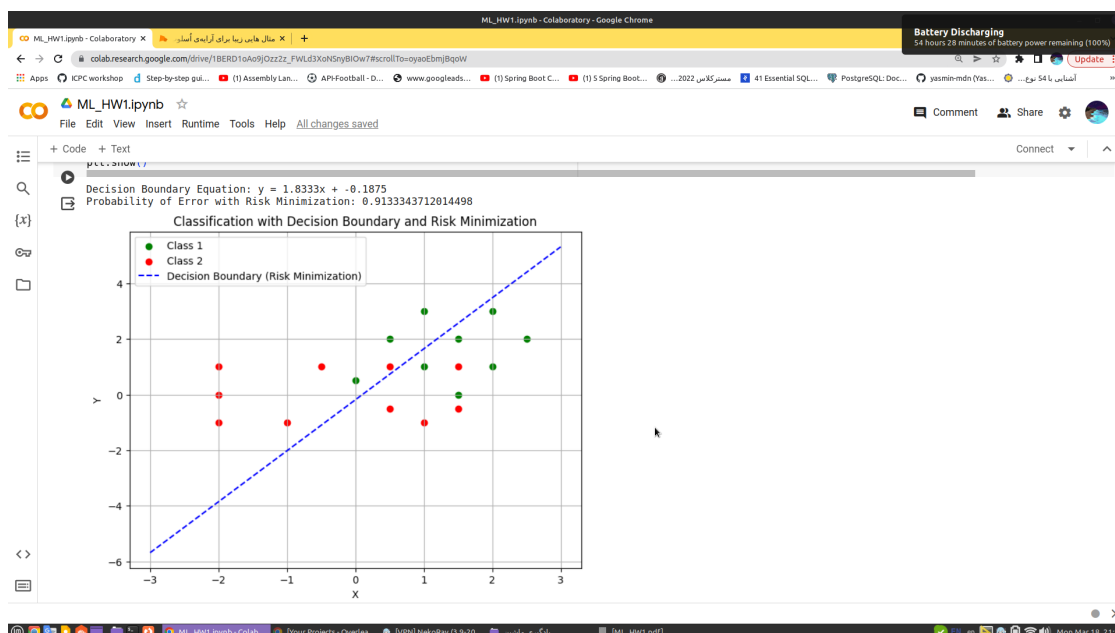
برای محاسبه خط مرز تصمیم، شیب و عرض از مبدا خط واصل بین میانگین دو توزیع گوسی را به دست خواهیم آورد. همانطور که از شکل زیر نیز مشخص است تقریباً ۹ نقطه از ۱۹ نقطه به اشتباه طبقه‌بندی شده‌است.



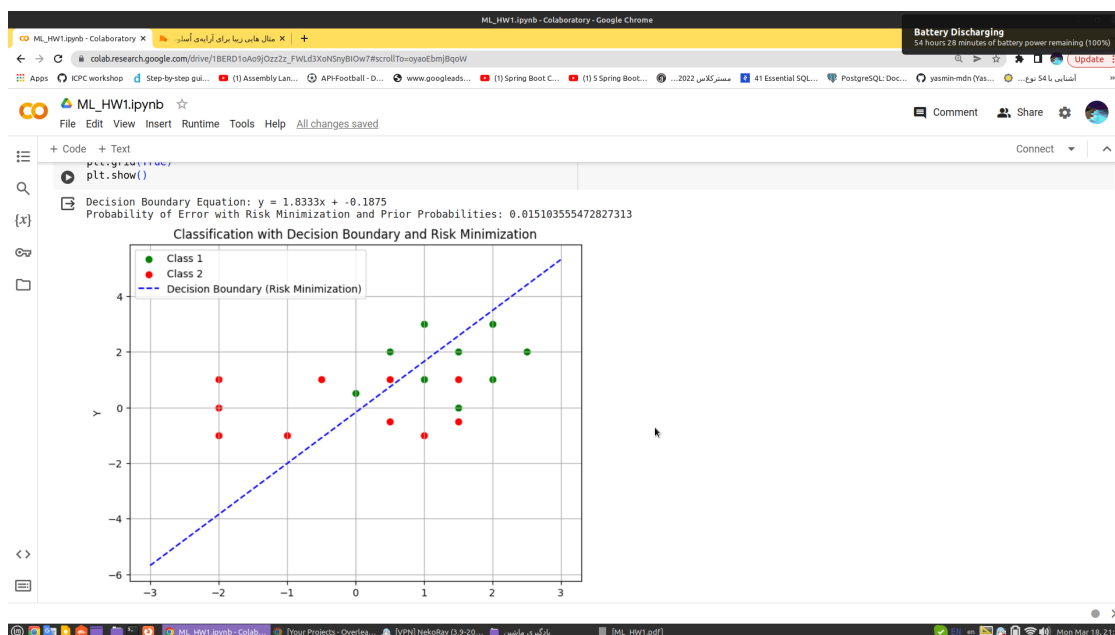
شکل ۶: محاسبه شیب و عرض از مبدأ خط مرز تصمیم

با توجه به مقداردهی های پارامتر ذکر شده در ماتریس ریسک بازهم به محاسبه خط مرز تصمیم و سپس شاهد آن خواهیم بود که ۷ نقطه از ۱۹ نقطه به اشتباه طبقه بندی شده است. متغیری که در تصویر نمایش داده شده است در واقع انتگرال مساحت ناحیه خطا می باشد.

نسبت به حالت قبل که از ماتریس ریسک استفاده نکردیم خطا به صورت خوبی کاهش پیدا نمود.



شکل ۷: رسم خط مرز تصمیم و ماتریس ریسک



شکل ۸: رسم خط مرز تصمیم با استفاده از ماتریس ریسک و احتمال پیشین



۶ تمرین ششم

پاسخ مربوط به این بخش در فایل Razavi ML HW1 810102155 نوشته شده است.

۷ تمرین هفتم

در این تمرین ابتدا می‌بایستی به پیش‌پردازش مجموعه داده بپردازیم ، سپس با استفاده از منطق احتمال شرطی بیز به محاسبه میزان تعلق هر یک داده‌ها به هر کدام از کلاس‌ها خواهیم پرداخت.

Naive Bayes یک مدل خاص از احتمال شرطی می‌باشد که این فرض را در نظر می‌گیرد ، که همه ویژگی‌ها feature از یکدیگر مستقل هستند.

تفاوت اصلی میان Naive Bayes و طبقه‌بندی بیزی در همین استقلال ویژگی‌ها از یکدیگر است. زمانی بهتر است از این مدل‌ها استفاده کنیم که شرط استقلال feature از یکدیگر درست باشند. همچنین اگر تعداد ابعاد زیاد باشد و کارایی در محاسبات بسیار مهم است. زمانی که استقلال میان ویژگی‌ها برقرار نیست ، استفاده از این مدل هزینه زیادی دارد. این مدل معروف به مدل high bias low variance می‌باشد.

```

print('Accuracy : ', accuracy)
# print("Confusion Matrix : ")
# print(conf_matrix)

Accuracy: 0.9516129032258065
Confusion Matrix:
[[ 1  1]
 [ 2 58]]

Classification Report:
              precision    recall  f1-score   support

     1           0.33         0.50         0.40         2
     2           0.98         0.97         0.97         60

 accuracy          0.95         0.95         0.95         62
 macro avg         0.66         0.73         0.69         62
 weighted avg      0.96         0.95         0.96         62

use SKLEARN

[ ] from sklearn.model_selection import train_test_split
    from sklearn.naive_bayes import GaussianNB
    from sklearn.metrics import accuracy_score, classification_report, confusion_matrix

def preprocess_data(data):
    # Convert categorical variables to numerical values
    data = data.replace({'W': 1, 'F': 2, 'YES': 2, 'NO': 1, 'x': 0})

    # Split data into features and labels

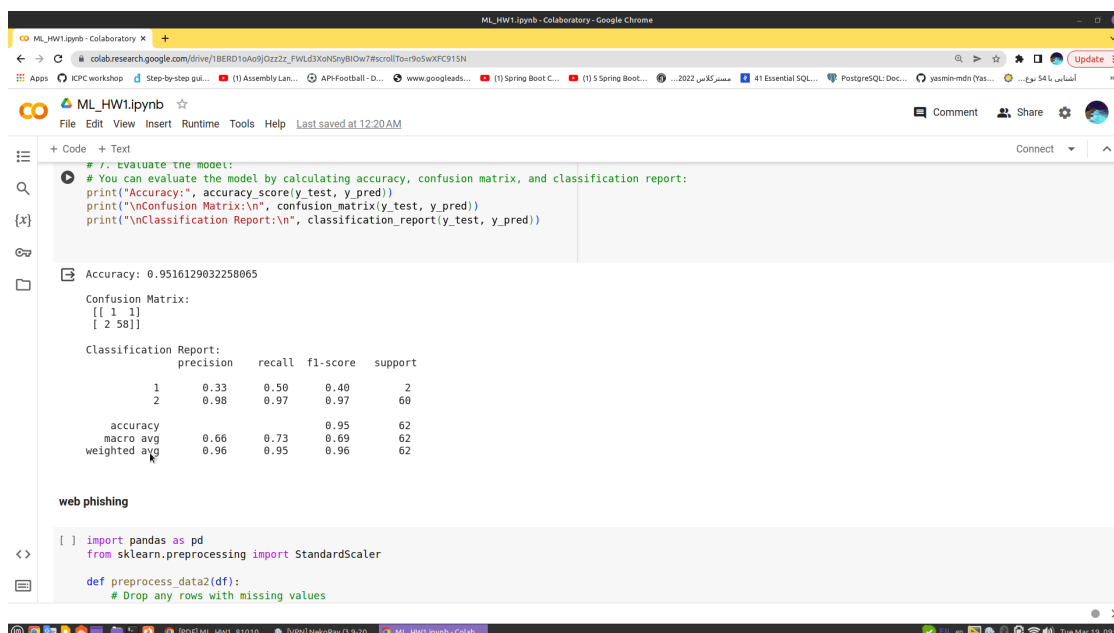
```

شکل ۹: پیاده‌سازی بدون کتابخانه از نایوبیز برای مجموعه داده سرطان ریه

[h] برای پیش‌پردازش داده‌ها ابتدا کاراکترها را با یک معادل عددی جایگزین خواهیم کرد ، سپس به پیاده‌سازی مدل خواهیم پرداخت. برای پیاده‌سازی این مدل نیازمند fit کردن داده‌های آموزشی هستیم. در این تابع بایستی احتمال پیشین و احتمال likelihood را محاسبه کنیم. سپس در تابع predict به محاسبه

$$P(X|W_1) <> P(X|W_2)$$

می‌پردازیم . هر کدام که مقدار بیشتری داشت را به عنوان برچسب کلاس انتخاب خواهیم کرد. می‌بینیم که این مدل مشابه پیاده‌سازی SKLEARN به محاسبه ماتریس آشفستگی می‌پردازد. تقریباً تفاوتی میان مدل ما و مدل پیش‌ساخته SKLEARN وجود ندارد.



```
# Evaluate the model:
# You can evaluate the model by calculating accuracy, confusion matrix, and classification report:
print("Accuracy:", accuracy_score(y_test, y_pred))
print("\nConfusion Matrix:\n", confusion_matrix(y_test, y_pred))
print("\nClassification Report:\n", classification_report(y_test, y_pred))
```

Accuracy: 0.9516129032258065

Confusion Matrix:

```
[[ 1  1]
 [ 2 50]]
```

Classification Report:

	precision	recall	f1-score	support
1	0.33	0.50	0.40	2
2	0.98	0.97	0.97	60
accuracy			0.95	62
macro avg	0.66	0.73	0.69	62
weighted avg	0.96	0.95	0.96	62

web phishing

```
[ ] import pandas as pd
from sklearn.preprocessing import StandardScaler

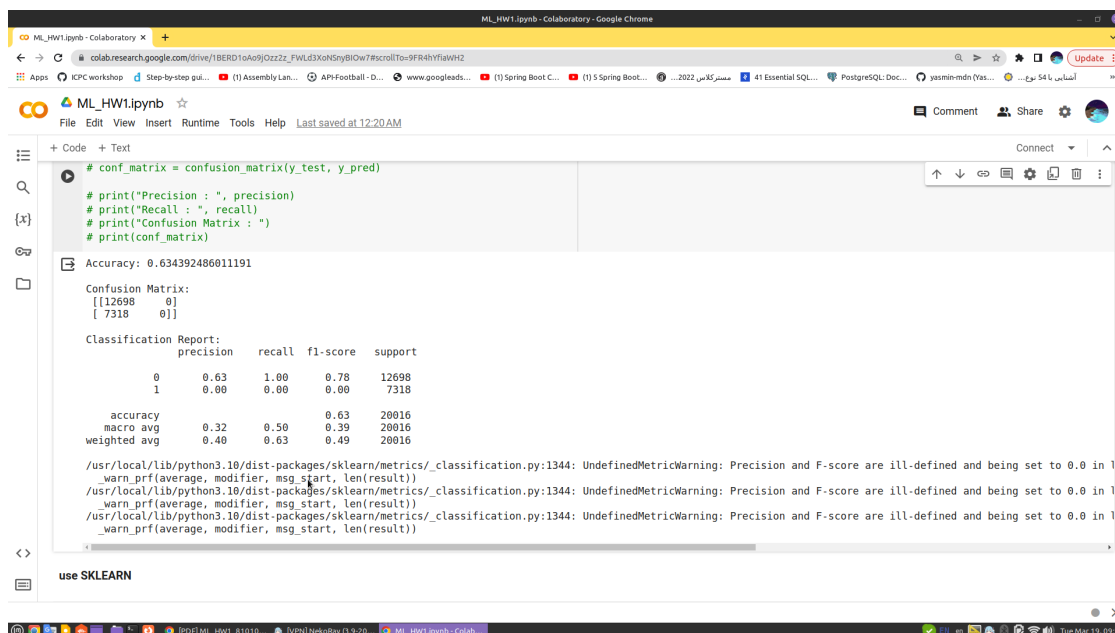
def preprocess_data2(df):
    # Drop any rows with missing values
```

شکل ۱۰: پیاده‌سازی SKLEARN از نایوبیز برای مجموعه داده سرطان ریه

حال به بررسی مجموعه داده دوم خواهیم پرداخت. این مجموعه داده، برخلاف مجموعه داده قبلی دارای ویژگی‌ها با مقادیر پیوسته می‌باشد.

همانطور که از مقایسه دقت‌ها نیز مشخص است، دقت مدل ما با اختلاف فاحشی کمتر از مدل پیش‌ساخته SKLEARN می‌باشد. به نظر می‌رسد که دقت مدل ما در تخمین احتمالات Likelihood دقت کمتری نسبت به مدل پیش‌ساخته SKLEARN دارد. علت این امر می‌تواند ناشی از چند عامل باشد.

از مرحله پیش‌پردازش می‌توان گذر کرد چرا که هر دو روش را با یک تابع پیش‌پردازش و نرمال‌سازی کرده‌ایم. شاید داده‌ها بر روی داده آموزشی overfit شده باشد در مدلی که ما ساخته‌ایم.



```
# conf_matrix = confusion_matrix(y_test, y_pred)

# print("Precision : ", precision)
# print("Recall : ", recall)
# print("Confusion Matrix : ")
# print(conf_matrix)

Accuracy: 0.634392486011191

Confusion Matrix:
[[12698  0]
 [ 7318  0]]

Classification Report:
              precision    recall  f1-score   support

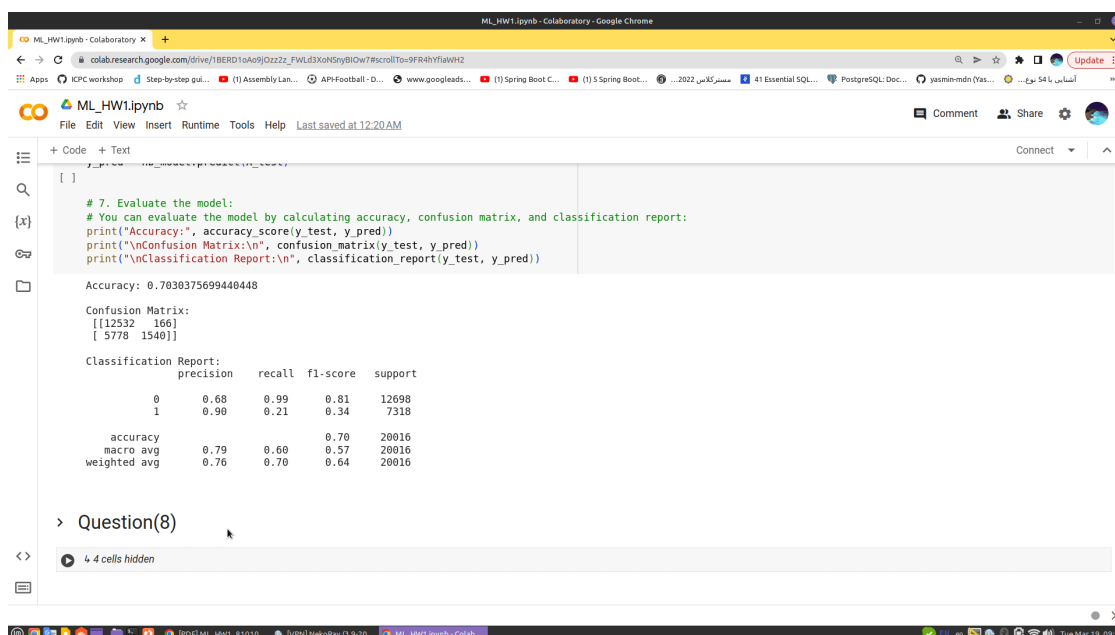
      0       0.63        1.00        0.78      12698
      1       0.00        0.00        0.00       7318

 accuracy          0.63          0.63          0.63      20016
 macro avg         0.32          0.50          0.39      20016
 weighted avg      0.40          0.63          0.49      20016

/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1344: UndefinedMetricWarning: Precision and F-score are ill-defined and being set to 0.0 in 1
_warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1344: UndefinedMetricWarning: Precision and F-score are ill-defined and being set to 0.0 in 1
_warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1344: UndefinedMetricWarning: Precision and F-score are ill-defined and being set to 0.0 in 1
_warn_prf(average, modifier, msg_start, len(result))

use SKLEARN
```

شکل ۱۱: پیاده‌سازی بدون کتابخانه از نایوبیز برای مجموعه‌داده وب



```
# 7. Evaluate the model:
# You can evaluate the model by calculating accuracy, confusion matrix, and classification report:
print("Accuracy:", accuracy_score(y_test, y_pred))
print("\nConfusion Matrix:\n", confusion_matrix(y_test, y_pred))
print("\nClassification Report:\n", classification_report(y_test, y_pred))

Accuracy: 0.7030375699440448

Confusion Matrix:
[[12532  166]
 [ 5778 1540]]

Classification Report:
              precision    recall  f1-score   support

      0       0.68        0.99        0.81      12698
      1       0.90        0.21        0.34       7318

 accuracy          0.70          0.70          0.70      20016
 macro avg         0.79          0.60          0.57      20016
 weighted avg      0.76          0.70          0.64      20016

> Question(8)

4 cells hidden
```

شکل ۱۲: پیاده‌سازی SKLEARN از نایوبیز برای مجموعه‌داده وب

۸ تمرین هشتم

برای این منظور می‌توانیم از معیارهایی چون میانگین پیکسل‌ها ، واریانس پیکسل‌ها و حتی مد پیکسل‌ها نیز استفاده کنیم.

ذکر این نکته را قبل از بررسی معیارهای فوق بر روی تصاویر الزامی می‌دانم که به نظر شخصی بنده بعضی تصاویر برجسب مناسبی ندارند. حداقل ایهام بالایی در تشخیص تصاویر موجود است.

برای این مساله اولاً ذکر این نکته در محاسبات ضرورت دارد که هر کجا برجسب داده با برجسب پیش‌بینی شده مطابقت داشت را به عنوان مقداری برای True Positive در نظر گرفته‌ایم.
مقادیر False Positive و True Negative نیز بر حسب اتفاق یکی از حالات اشتباه در نظر گرفته‌ایم.
در هر کدام از ۳ طبقه‌بند زیر یک مقداری را بر حسب مشاهداتی که از این ۸۰ فایل داشتیم ، انتخاب کرده و سپس آن را به عنوان threshold در نظر می‌گیریم.

```

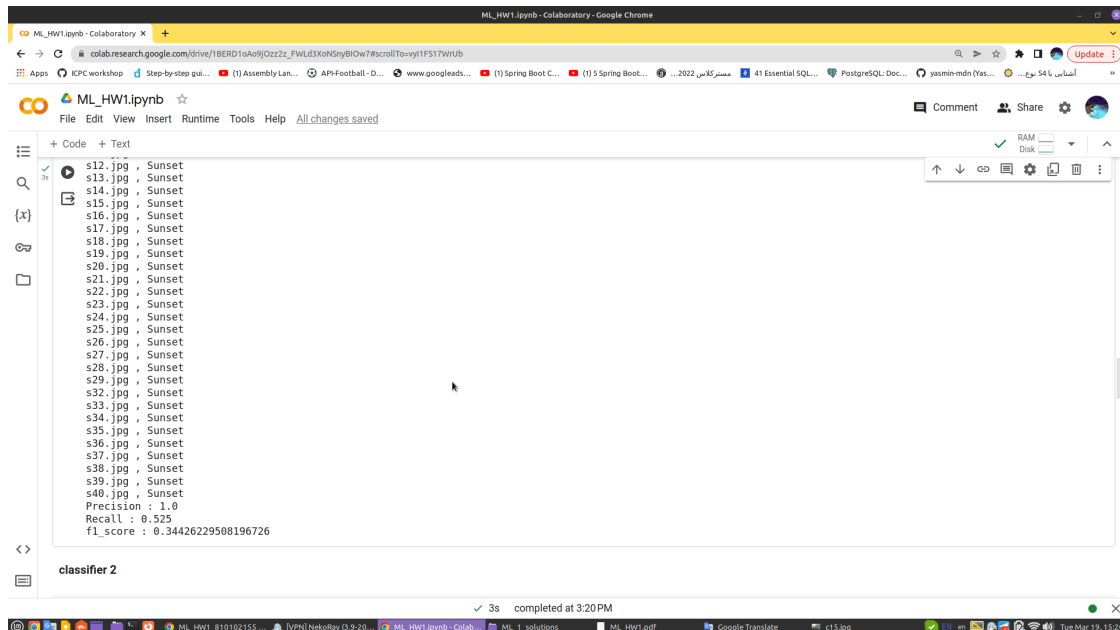
c6.jpg , Cloudy
c7.jpg , Cloudy
c9.jpg , Cloudy
c10.jpg , Cloudy
c11.jpg , Cloudy
c13.jpg , Cloudy
c14.jpg , Cloudy
c16.jpg , Cloudy
c17.jpg , Cloudy
c18.jpg , Cloudy
c21.jpg , Cloudy
c22.jpg , Cloudy
c24.jpg , Cloudy
c26.jpg , Cloudy
c27.jpg , Cloudy
c29.jpg , Cloudy
c30.jpg , Cloudy
c32.jpg , Cloudy
c33.jpg , Cloudy
c35.jpg , Cloudy
c36.jpg , Cloudy
c37.jpg , Cloudy
c38.jpg , Cloudy
c39.jpg , Cloudy
c40.jpg , Cloudy
Precision : 0.6486486486486487
Recall : 0.8888888888888888
f1_score : 0.375

classifier 1

[31] # https://drive.google.com/file/d/1YmP6MScAZwv28Jsn5Qsdsjb15tMidD8z/view?usp=drive_link
from oooole.colab import drive
  
```

شکل ۱۳: نتایج طبقه‌بندی بر اساس میانگین پیکسل‌ها

به عنوان مثال طبقه‌بند بر اساس پیکسل‌ها توانسته‌است یک کلاس را به صورت کامل درست تشخیص دهد اما در تشخیص کلاس غروب به سختی چند مورد را تشخیص داده‌است.



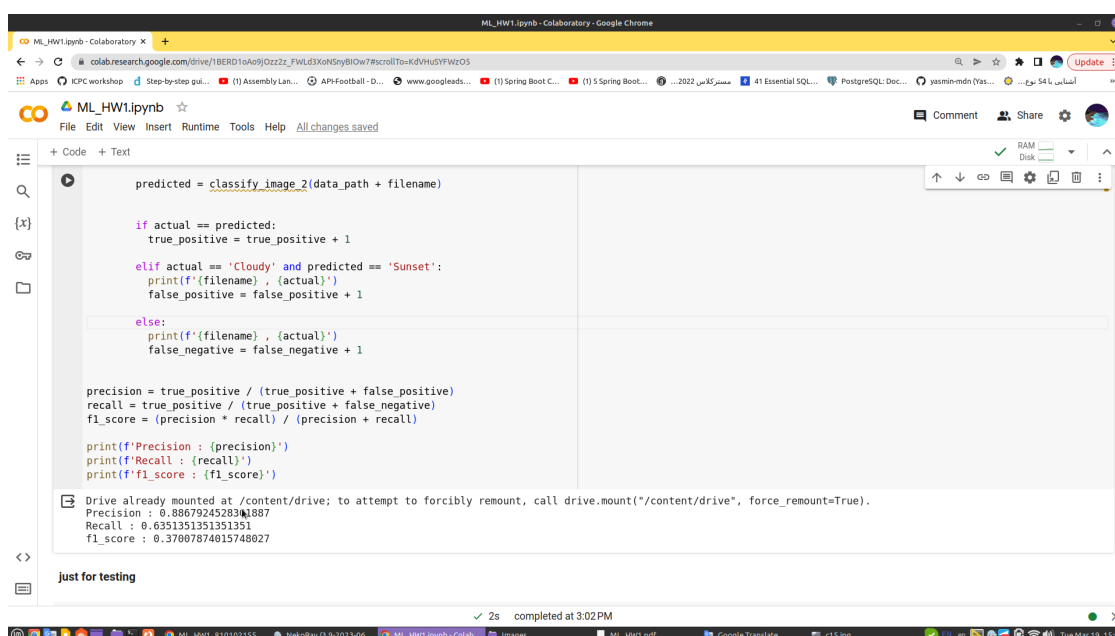
شکل ۱۴: نتایج طبقه‌بندی بر اساس واریانس پیکسل‌ها

از محاسبات دقت و بازسازی اینطور به نظر می‌رسد که معیار مد پیکسل‌ها تا حد خوبی توانسته‌است که جداسازی میان تصاویر را انجام دهد.
(تصور شخصی من اینطور بود که میانگین پیکسل‌های تصاویر ابری باید به شدت متفاوت از تصاویر غروب باشد، اما به نسبت عالی عمل نکرد این معیار)
همچنین اینطور به نظر می‌رسد که پیکسل با فراوانی بیشتر که همان مد خوانده می‌شود توانسته حدفاصل خوبی برای ایجاد اختلاف بین این دو کلاس از تصاویر ایجاد کند.

به عنوان مثالی دیگر معیار میانگین برای عکس s8 به اشتباه عمل کرده‌است. و این امر طبیعی به نظر می‌رسد، چرا که میزان ابرهای درون تصویر باعث شده است که میانگین پیکسل‌ها در حدود میانگین تصاویر ابری باشد.

معیار واریانس برای هیچ‌کدام از تصاویر غروب به خوبی عمل نکرده‌است. میزان پراکندگی در این داده‌ها به طرز چشمگیری مشهود است. همچنین یک بخشی از این اتفاق به نظر من به خاطر برجسته شدن بعضی از داده‌ها می‌باشد.

در مورد شاخص مد نیز به عنوان مثال، تصویر s30 که در این تصویر بیشترین فراوانی مربوط به رنگ سفید و ابرها می‌باشد. به همین ترتیب تصاویر s8 و s9.



```
predicted = classify_image_2(data_path + filename)

if actual == predicted:
    true_positive = true_positive + 1

elif actual == 'Cloudy' and predicted == 'Sunset':
    print(f'{filename} , {actual}')
    false_positive = false_positive + 1

else:
    print(f'{filename} , {actual}')
    false_negative = false_negative + 1

precision = true_positive / (true_positive + false_positive)
recall = true_positive / (true_positive + false_negative)
f1_score = (precision * recall) / (precision + recall)

print(f'Precision : {precision}')
print(f'Recall : {recall}')
print(f'f1_score : {f1_score}')

Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).
Precision : 0.8867924532831887
Recall : 0.6351351351351351
f1_score : 0.37887874015748927
```

just for testing

2s completed at 3:02PM

شکل ۱۵: نتایج طبقه‌بندی بر اساس مد پیکسل‌ها