به نام خدا

گزارش تمرین سری دوم – قسمت دوم درس داده کاوی

نام و نام خانوادگی: پویا شاعری

شماره دانشجویی: ۴۰۰۴۲۲۱۰۵

لینک Colab:

https://colab.research.google.com/drive/1VRfoc9z5nL6BDW4t4vRt5gIU2vCsnJWa?usp=sharing

چکیده

در این گزارش قصد داریم مرحله به مرحله کارهای انجام شده را توضیح دهیم و تحلیل کنیم. گرچه در داخل نوت‌بوک، به صورت تکست بالای هر سلول توضیح داده شده است ولی ضروری است توضیحات و تحلیل های لازم و مقایسه ها را در اینجا داشته باشیم.

مقدمه

در این سری از تمارین به دلیل ایجاد تفاوت با سری پیش و در دسترس نبودن دیتاست قسمت دوم تصمیم گرفتیم که دیتاست را از گوگل‌درایو، mount کنیم که این کار را انجام دادیم.

پیش از هرچه به پیش پردازش داده ها می‌پردازیم (Task 0) ، تعداد داده‌های میسینگ برای هر فیچر برابر صفر بوده و برای آوتلایر دیتکشن به صورت دستی z-score زدیم و از ۳۰۳ رکورد به ۲۸۷ رسیدیم. حال با یک دیتافریم تمیز سر و کار داریم، گرچه این دیتاست، بنچمارک بوده و با توجه به میسینگ نداشتن و ویژگی های خوبی نظیر کم تعداد بودن کار با آن نسبتا آسان است.

سوال اول

برای محاسبه احتمال وقوع یک رخداد A به شرط وقوع رخداد B یعنی $P(A|B)$ معادله بیز برای احتمال شرطی با درنظر گرفتن $P(B) > 0$ به‌صورت زیر برقرار است:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

به طرف راست، احتمال پسین می‌گویند. همچنین قسمت اول صورت کسر نیز، تابع (likelihood_function) و قسمت آخر صورت کسر هم احتمال پیشین نامیده می‌شود.

حال اگر فضای نمونه توسط $B_1, B_2, \dots, B_j$ افراز شده باشد بطوری‌که احتمال هر کدام از این افرازها، بزرگتر اکید از صفر باشد، در این صورت برای هر پیشامد A داریم :

$$P(B_j|A) = \frac{P(B_j)P(A|B_j)}{P(A)}$$

با جایگذاری رابطه احتمال شرطی و قانون ضرب احتمال، داریم:

$$P(B_j|A) = \frac{P(B_j)P(A|B_j)}{\sum_{i=1}^{n} P(B_i)P(A|B_i)}$$

به این قضیه در ماشین لرنینگ، قضیه بیز می‌گوییم. نکته‌ای که حائز اهمیت است، این است که شرط بیز برای احتمال پیشامد، ضعیف تر از شرط استقلال پیشامد هاست.

دسته‌بندی روش بیز در اغلب موارد به عنوان یک راهکار ساده برای دسته‌بندی و تعیین تشخیص برچسب اشیاء استفاده می‌گردد. اما برای بکارگیری دسته بندی بیز ساده، الگوریتم خاصی وجود ندارد، ولی در عوض خانواده ای از الگوریتم‌ها موجود است که با فرض استقلال و یژگی‌ها یا متغیرها نسبت به یکدیگر عمل می‌نمایند. اما دسته بند بیز ساده چندجمله‌ای یا Multinomial Naive Bayes به عنوان یک دسته بندی متنی مورد استفاده است که برحسب مدل احتمالی یا توزیع چندجمله ای، برداری از ویژگی ها در نظر می گیرد که برای ویژ گی هایی که ارائه دهنده اعداد گسسته هستند، مناسب تر است. ولی دسته بندی بیز ساده برنولی Bernoulli Naive

Bayes که شبیه بیز ساده چند جمله ای است، اما فرضیات متغیرهای بولی هستند. در آن فرض می شود ویژگی ها دودویی باشند (صفر و یک) و به شکلی دسته بندی بیز را ایجاد می‌نماید که بیشترین کاربرد را در دسته بندی متن‌های کوتاه دارد، به همین دلیل محبوبیت و دارای کاربرد بیشتری می باشد.

سوال دوم و سوم

دسته بند Gaussian Naive Bayes یک الگوریتم احتمالی (Probabilistic) بوده و شامل محاسبه احتمال پیشین و پسین اعضای کلاس بودن دیتاست ترین و متعاقبا تست است.
احتمالات مشروط ویژگی‌های داده‌های تست با یک کلاس داده می‌شود. احتمالات مشروط ویژگی‌های داده آزمایشی که یک کلاس داده می‌شود با احتمال به‌دست‌آمده از توزیع گاوسی (نرمال) داده می‌شود.

$$P(x_i|c) = \frac{1}{\sqrt{2\pi}\,\sigma}\; e^{\left(\frac{-(x_i-\mu)^2}{2\sigma^2}\right)}$$

در نهایت، احتمال شرطی هر کلاس با یک نمونه با استفاده از قضیه بیز محاسبه می‌شود.

$$P(c_i|x) = \frac{P(x|c_i)P(c_i)}{\sum_{j=1}^{n} P(x|c_j)P(c_j)}$$

تمام نتایج در کد فایل به همراه کامنت و توضیح بالای هر سلول داده شده و به طور خلاصه برخی نتایج را آورده ایم:

```
accuracy = calculate_accuracy(x_test, predictions)
print("Accuracy : ", accuracy)
```

```
Accuracy :  72.41379310344827
```

```
from sklearn.metrics import precision_score, recall_score, f1_score

y_pred = [ value for key,value in predictions.items()]

print("Precision:")
my_p_score = precision_score(y_test, y_pred)
my_p_score * 100
```

```
Precision:
75.67567567567568
```

```
print("Recall:")
my_recall_score = recall_score(y_test, y_pred)
my_recall_score * 100
```

```
Recall:
80.0
```

```
print("F1 Score:")
my_f1_score = f1_score(y_test, y_pred)
my_f1_score * 100
```

```
F1 Score:
77.77777777777779
```

سوال چهارم و پنجم

تمام نتایج در کد فایل به همراه کامنت و به طور خلاصه برخی نتایج را آورده ایم:

```
[15]  1 accuracy = accuracy_score(y_test, sk_y_pred)
      2 print("Accuracy : ", accuracy * 100)
```

```
Accuracy :  72.41379310344827
```

```
 1 print("Precision:")
 2 p_score = precision_score(y_test, sk_y_pred)
 3 print(p_score * 100)
 4
 5 print("\nMy Precision:")
 6 print(my_p_score * 100)
```

```
Precision:
67.56756756756756

My Precision:
75.67567567567568
```

```
[17]  1 print("Recall:")
      2 recall_score = recall_score(y_test, sk_y_pred)
      3 print(recall_score * 100)
      4
      5 print("\nMy Recall:")
      6 print(my_recall_score * 100)
```

```
Recall:
86.20689655172413

My Recall:
80.0
```

```
 1 print("F1 Score:")
 2 f1_score = f1_score(y_test, sk_y_pred)
 3 print(f1_score * 100)
 4
 5 print("\nMy F1 Score:")
 6 print(my_f1_score * 100)
```

```
F1 Score:
75.75757575757575

My F1 Score:
77.77777777777779
```