دانشکدۀ مهندسی صنایع

پیاده‌سازی یک سیستم پیش‌بینی وام روی داده‌های مالی

دانشجویان :

شقایق حیدری
شهرزاد مهرنورد

استاد درس :

دکتر غضنفری

تیر ماه 1401

چکیده:

در این پروژه هدف ما این است که بفهمیم اگر مقداری اطلاعات مالی از مشتریان داشته باشیم، و به آنها وام بدهیم، آیا وام را به ما برمی‌گردانند یا خیر؟

به این منظور از یک دیتاست وام استفاده کردیم و پس از طی مراحل جستجو و پیش پردازش داده‌ها دو الگوریتم درخت تصمیم و درخت تصادفی را روی آن پیاده سازی کردیم که در نهایت نتایج بدست آمده از درخت تصادفی قابل توجه بود.

فهرست محتوا

مقدمه

به تازگی سه اقتصاد دان به نام های آدد نزار ، آلان لامر و مایکل هرسنشتاین ، از دانشگاه Delaware ، به دنبال راه هایی گشته اند که با استفاده از آنها بشود امکان پس دادن یا ندادن وام از طرف وام گیرندگان را احتمال سنجی کرد . در این راه ، در مجموعه داده هایی که آنها استفاده کرده اند ، وام گیرندگان معمولا یک توضیح مختصر درباره ی دلیل وام گرفتن خود و اینکه تخمین میزنند چه زمان وام خود را باز پرداخت کنند ، نوشته بودند . وام دهندگان هم بر اساس همین توضیحات تصمیم میگرفتند که به این افراد وام بدهند یا خیر .

کاری که درواقع در این مجموعه داده ها انجام شده بود ، استفاده از زبان و نوع جمله بندی و کلمات به کار برده شده توسط وام گیرندگان بود که از نظر محققان مشخص میکرد هر فرد تا چه حد احتمال خوش قولی دارد و چقدر احتمال دارد وام خود را پس بدهد . در این مجموعه داده ها ، کلمات به کار رفته در جملات هر فرد آنالیز شده و طبق آن به نتیجه میرسیدند که آیا این فرد خوش قول خواهد بود و وام خود را پس خواهد داد یا خیر که البته تا حد معمولی به نتیجه رسیده بودند .

اما بعد از مدتی این سوال اخلاقی پیش آمد که با اینکه این کار یک کار تحقیقاتی است اما چقدر میتوانیم افراد را از روی کلمات و نوع جمله بندی آنها قضاوت کنیم ؟ در این صورت افراد نیازمند به وام نه تنها باید نگران وضعیت مالی و بانکی خود باشند ، بلکه باید نگران تاریخچه ی متون و جملاتی که نوشته اند هم باشند تا مبادا این جملات مانع همکاری وام دهندگان با آنها شود .

بنابراین با انفجار شهرت تحلیل داده و کلان داده ، استفاده از چنین متد هایی زیاد مورد استقبال قرار نمیگیرد . همینطور کلان داده و تحلیل داده ها این موضوع که آیا میشود تنها از روی کلمات انتخابی که یک شخص در توضیح دلایل خود برای گرفتن وام ، او را قضاوت اخلاقی کرد ، زیر سوال میبرد .

امروزه میتوان از داده هایی که درباره ی تجربه های گذشته ی اشخاص وجود دارد ، استفاده و تصمیم گیری ها را آسان تر کرد . برای مثال خیلی از شرکت ها از تحلیل داده های تاریخچه ی کاری یک فرد میتوانند به نتیجه برسند که آیا این فرد برای شرکت آنها مفید خواهد بود و استخدام کردن او کار درستی است یا خیر . در این مورد هم میتوان از آنالیز داده و کلان داده برای تصمیم گیری درباره ی اینکه چقدر احتمال دارد یک فرد وامی را که گرفته به موقع پس بدهد ، تصمیم گیری کرد .

بنابراین استفاده از داده های موجود درباره ی افراد میتواند به ما کمک کند تا بدانیم به چه افرادی میتوان اعتماد کرد و این موضوع نه تنها در احتمال پس گیری وام ، بلکه در بسیاری موارد دیگر ، قابل استفاده است.

در این پروژه ، مجموعه ی داده ای که استفاده شده ، مرتبط با شرکت LendingClub است که یک شرکت وام دهنده ی همتا به همتا بوده که دفتر مرکزی آن در سانفرانسیسکو، کالیفرنیا قرار دارد. این شرکت اولین وام دهنده ی همتا به همتا بود که پیشنهادات خود را به عنوان اوراق بهادار در کمیسیون بورس و اوراق بهادار ثبت کرد و معاملات وام را در بازار ثانویه ارائه داد.

در این پروژه میخواهیم با توجه به اطلاعاتی که از مشتریان LendingClub موجود است ، به نتیجه برسیم که کدام یک از آنها به احتمال قوی تر وام خود را پس میدهند که بنابراین میتوانند در لیست وام گیرندگان این شرکت قرار بگیرند .

شرایط صلاحیت داشتن برای گرفتن وام از شرکت lendingclub

در شرکت LendingClub ، 3 شرط اصلی برای گرفتن وام وجود دارد :

1- شهروند ایالات متحده یا مقیم دائم آن باشید یا با ویزای معتبر و طولانی مدت در ایالات متحده زندگی کنید .
2- حداقل 18 سال را داشته باشید .
3- یک حساب بانکی قابل تایید داشته باشید .

شرایط قبول شدن درخواست وام (فاکتور هایی که موقع بررسی درخواست ها به آنها توجه میشود):

1- اطلاعات کلی ذکر شده داخل درخواست نامه
2- اطلاعات مربوط به شما از دفاتر اعتباری
3- امتیاز اعتباری شما
4- سایر اطلاعاتی که احتمال پرداخت به موقع تا زمان بازپرداخت کامل وام را پیش بینی میکند.

از نظر شرکت lendingclub درخواست نامه هایی که شرایط زیر را دارند ، در اولویت میباشند :

1- امتیاز اعتباری بالا
2- درصد پایین بدهی نسبت به مبلغ درآمد
3- سابقه طولانی مدت خطوط اعتباری موفق

فاز اول

Import Libraries:

ابتدا کتابخانه های مورد نیاز را ایمپورت می‌کنیم.

```
In [1]: import pandas as pd
        import numpy as np
        import matplotlib.pyplot as plt
        import seaborn as sns
```

فاز دوم

Get the Data:

دیتاست مورد نظر را فرخوانی میکنیم و اطلاعات کلی و آماری آن را نمایش می‌دهیم.

```
In [3]: df = pd.read_csv('loan_data.csv')
```

```
In [3]: df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9578 entries, 0 to 9577
Data columns (total 14 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   credit.policy      9578 non-null   int64
 1   purpose            9578 non-null   object
 2   int.rate           9578 non-null   float64
 3   installment        9578 non-null   float64
 4   log.annual.inc     9578 non-null   float64
 5   dti                9578 non-null   float64
 6   fico               9578 non-null   int64
 7   days.with.cr.line  9578 non-null   float64
 8   revol.bal          9578 non-null   int64
 9   revol.util         9578 non-null   float64
 10  inq.last.6mths     9578 non-null   int64
 11  delinq.2yrs        9578 non-null   int64
 12  pub.rec            9578 non-null   int64
 13  not.fully.paid     9578 non-null   int64
```

ملاحظه می‌کنیم که دیتاست ما داده ی خالی ندارد.

```
In [4]: df.describe()
```

Out[4]:

	credit.policy	int.rate	installment	log.annual.inc	dti	fico	days.with.cr.line	revol.bal	revol.util	inq.last.6mths	delinq.2yrs
count	9578.000000	9578.000000	9578.000000	9578.000000	9578.000000	9578.000000	9578.000000	9.578000e+03	9578.000000	9578.000000	9578.000000
mean	0.804970	0.122640	319.089413	10.932117	12.606679	710.846314	4560.767197	1.691396e+04	46.799236	1.577469	0.163708
std	0.396245	0.026847	207.071301	0.614813	6.883970	37.970537	2496.930377	3.375619e+04	29.014417	2.200245	0.546215
min	0.000000	0.060000	15.670000	7.547502	0.000000	612.000000	178.958333	0.000000e+00	0.000000	0.000000	0.000000
25%	1.000000	0.103900	163.770000	10.558414	7.212500	682.000000	2820.000000	3.187000e+03	22.600000	0.000000	0.000000
50%	1.000000	0.122100	268.950000	10.928884	12.665000	707.000000	4139.958333	8.596000e+03	46.300000	1.000000	0.000000
75%	1.000000	0.140700	432.762500	11.291293	17.950000	737.000000	5730.000000	1.824950e+04	70.900000	2.000000	0.000000
max	1.000000	0.216400	940.140000	14.528354	29.960000	827.000000	17639.958330	1.207359e+06	119.000000	33.000000	13.000000

```
In [5]: df.head()
```

Out[5]:

lit.policy	purpose	int.rate	installment	log.annual.inc	dti	fico	days.with.cr.line	revol.bal	revol.util	inq.last.6mths	delinq.2yrs	pub.rec	not.fully.paid
1	debt_consolidation	0.1189	829.10	11.350407	19.48	737	5639.958333	28854	52.1	0	0	0	0
1	credit_card	0.1071	228.22	11.082143	14.29	707	2760.000000	33623	76.7	0	0	0	0
1	debt_consolidation	0.1357	366.86	10.373491	11.63	682	4710.000000	3511	25.6	1	0	0	0
1	debt_consolidation	0.1008	162.34	11.350407	8.10	712	2699.958333	33667	73.2	1	0	0	0
1	credit_card	0.1426	102.92	11.299732	14.97	667	4066.000000	4740	39.5	0	1	0	0

هر کدام از فیچر های بالا معرف مشخصات زیر می باشند:

credit.policy : اگـر مشـتری تعهـد معیـار هـای مـرتبط بـا LandingClub.com را رعایـت کرده باشد ، 1 و در غیر این صورت ، 0 خواهد بود .

Purpose : هـدف وام گرفتـه شـده (مقـادیر " credit_card", "debt_consolidation", "educational"، "all_other" and "small_business", "major_purchase" را میگیرد)

int.rate : نـرخ بهـره ی وام (نـرخ 11٪ بـه صـورت 0.11 ذخیـره میشـود) . وام گیرنـدگانی کـه از نظـر LandingClub.com ، ریسک بیشتری دارند ، نرخ بهره ی بیشتری به آنها تعلق گرفته است .

installment : قسـط هـای ماهانـه ای کـه بـه وم گیرنـدگان در صـورت تـامین شـدن آن ، تعلـق میگیرد.

log.annual.inc : لاگ های طبیعی گزارش شده از درآمد سالانه ی وام گیرندگان .

dti : نسبت بدهی به درآمد وام گیرندگان (میزان بدهی تقسیم شده بر درآمد سالانه)

fico : امتیاز اعتباری FICO وام گیرنده .

days.with.cr.line : تعداد روز هایی که وام گیرنده ، خط اعتبار داشته است .

revol.bal : تعادل گردان وام گیرنده (میزان خط اعتباری استفاده شده مرتبط با کل اعتبار در دسترس)

inq.last.6mths : تعداد استعلامات وام گیرنده توسط طلبکاران در 6 ماه گذشته .

delinq.2yrs : تعداد دفعاتی که وام گیرنده بیش از 30 روز ، در 2 سال گذشته ، از پرداخت عقب افتاده است.

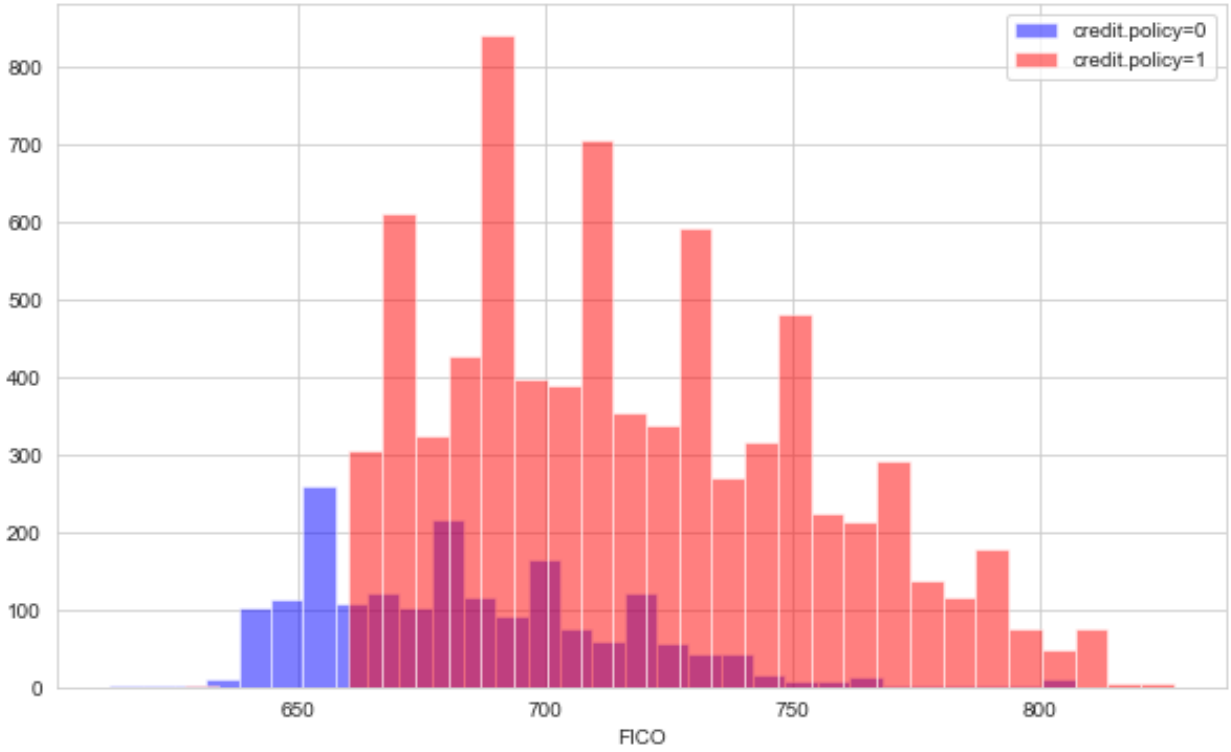
pub.rec : تعداد سوابق عمومی منفی وام گیرنده (پرونده های ورشکستگی، حق التزام مالیاتی، یا قضاوت)

فاز سوم

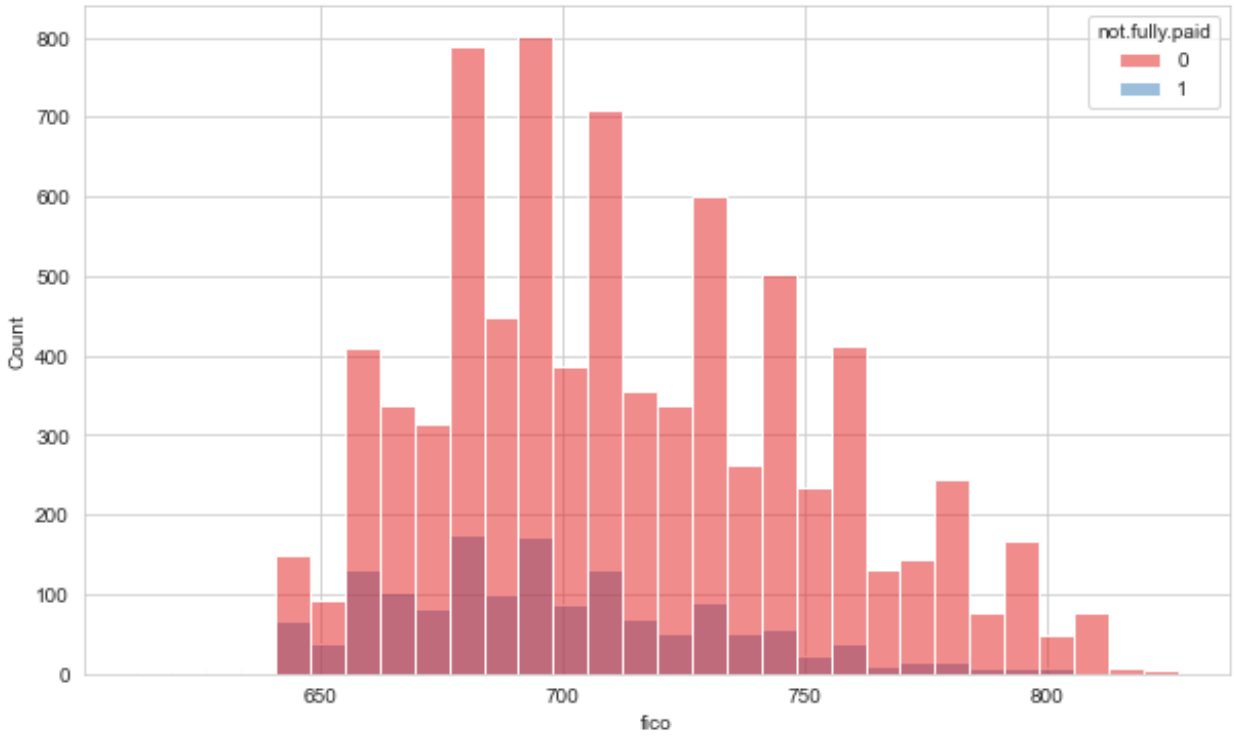
Exploratory Data Analysis:

حالا کمی در دیتاست گشت و گذار کنیم (EDA) تا ببینیم به چه اطلاعاتی دست پیدا می کنیم.

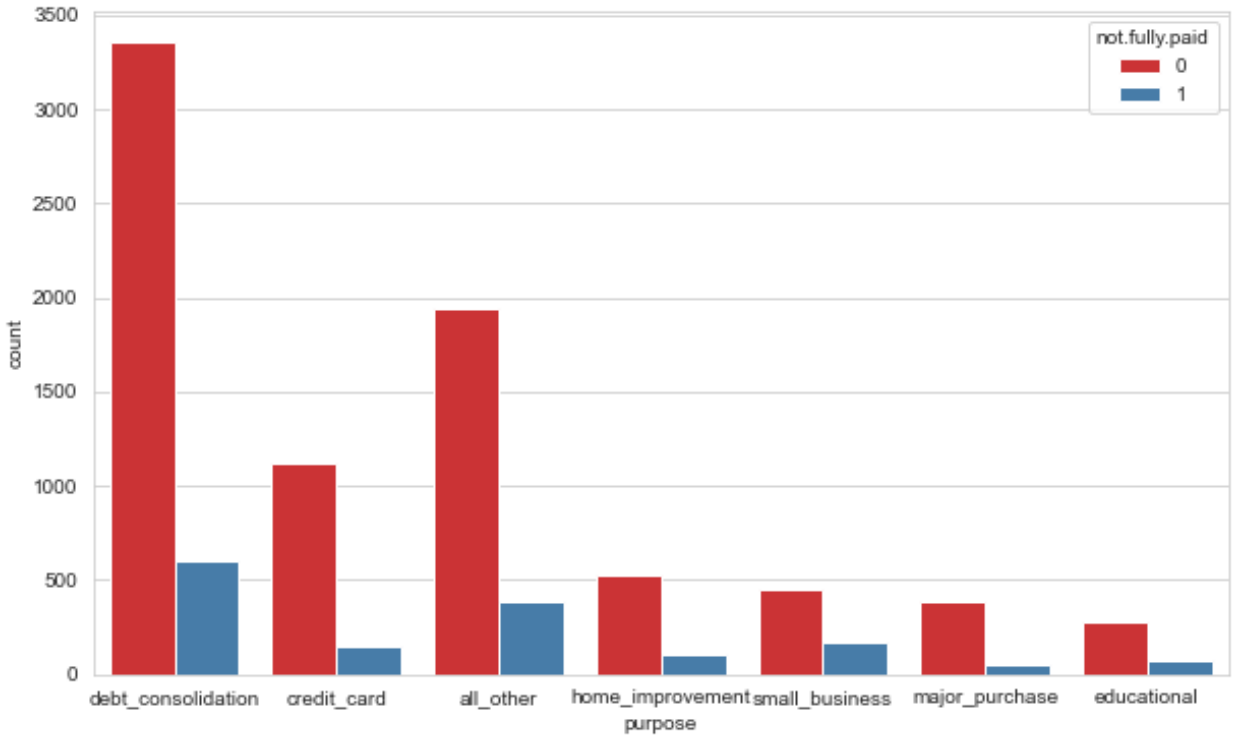
هیستوگرام فیچر fico را با تفکیک credit.policy در قالب دو نمودار روی هم ترسیم می‌نماییم.



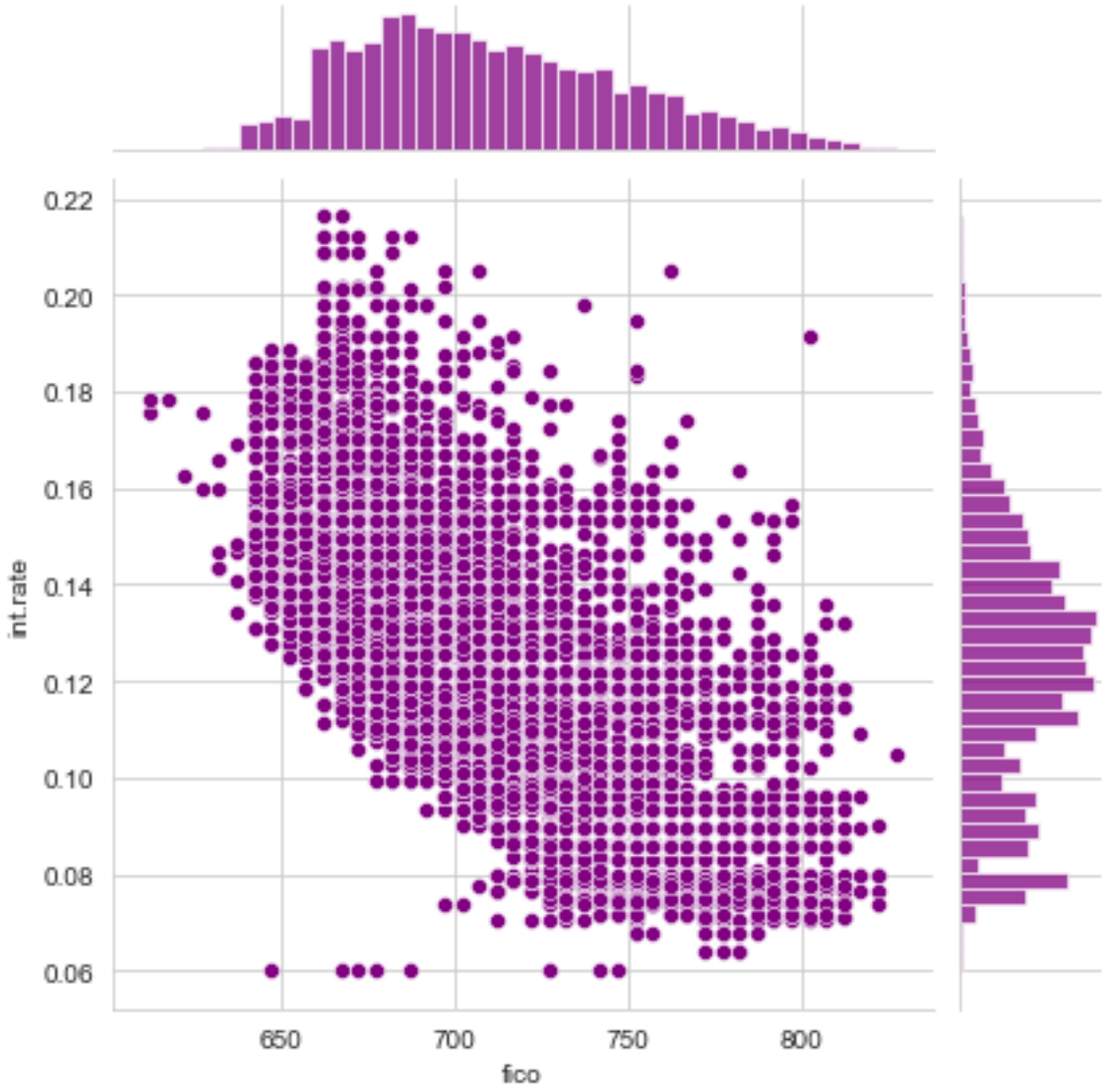
همان نمودار بالا را این بار با تفکیک لیبل یا همان not.fully.paid ترسیم می‌کنیم.



نمودار countplot را برای فیچر purpose با تفکیک مشتریانی که موفق به بازپرداخت وام شدند یا نشدند ترسیم می‌کنیم و ملاحظه می‌کنیم که اکثر افرادی که موفق به بازپرداخت وام خود نشده اند وام را به منظور تلفیق بدهی‌های قبلی دریافت کرده اند.



نمودار جوینت پلات را بر حسب دو فیچر fico و int.rate ترسیم می‌کنیم.



به نظر می آید که یک همبستگی کشف کردیم ، برای بررسی دقیق تر این همبستگی ها ، نمودار lmplot بر حسب fico و با دو ستون مختلف بر اساس این که وام پرداخت شده یا خیر و با تفکیک credit.policy رسم می‌کنیم.

نتیجه‌ای که میگیریم این است که : هر چه میزان ریسک یک مشتری کمتر باشد، امتیاز Fico مشتری بیشتر خواهد بود . این مسئله برای مشتریانی که موفق به پرداخت وام شده یا نشده اند و همچنین آنهایی که واجد شرایط اخذ وام بوده اند یا نه ، بررسی میشود.

فاز چهارم

Setting up the Data:

حـال در ادامـه تنظیمـاتی را روی دیتاسـت انجـام مـی‌دهیم تـا قابـل پـردازش شـود بـه عبـارتی بخشـی از پیش پردازش را اجرا میکنیم.

1. فیچر های categorical را به numerical تبدیل می کنیم.

بـا ذکـر جزئیـات بیشـتر درواقـع مـا سـتون purpose را بـه تفکیـک اهـداف آن numeric کردیم.

```
In [11]: pd.get_dummies(df,drop_first=True)
```

	fico	days.with.cr.line	revol.bal	revol.util	inq.last.6mths	delinq.2yrs	pub.rec	not.fully.paid	purpose_credit_card	purpose_debt_consolidation	purpose_educational	purpose_
3	737	5639.958333	28854	52.1	0	0	0	0	0	1	0	
4	707	2760.000000	33623	76.7	0	0	0	0	1	0	0	
5	682	4710.000000	3511	25.6	1	0	0	0	0	1	0	
6	712	2699.958333	33667	73.2	1	0	0	0	0	1	0	
7	667	4066.000000	4740	39.5	0	1	0	0	1	0	0	
...	...	...	...	...	...	...	...	...	...	...	...	
9574	672	10474.000000	215372	82.1	2	0	0	1	0	0	0	
9575	722	4380.000000	184	1.1	5	0	0	1	0	0	0	
9576	687	3450.041667	10036	82.9	8	0	0	1	0	1	0	
9577	692	1800.000000	0	3.2	5	0	0	1	0	0	0	
9578	732	4740.000000	37879	57.0	6	0	0	1	0	1	0	

فاز پنجم

Train Test Split:

گام بعد مرحله آموزش مدل است.

به این منظور داده های مان را به دو بخش آموزش و تست تقسیم می کنیم.

```
In [13]: from sklearn.model_selection import train_test_split
```

```
In [14]: X_train, X_test, y_train, y_test =
         train_test_split(df1.drop(['not.fully.paid'],axis=1),
                          df1['not.fully.paid'], test_size=0.30, random_state=101)
```

فاز ششم – قسمت اول

Training a Decision Tree Model:

اکنون با توجه به اینکه دیتاست ما از نوع labled است میتوانیم از الگوریتم درخت تصمیم برای تحلیل خود کمک بگیریم.

برای این منظور الگوریتم خود را به مدل آموزش می‌دهیم و آن را روی داده‌های آموزشی فیت میکنیم.

```
In [69]: from sklearn.tree import DecisionTreeClassifier
```

```
In [70]: dtree = DecisionTreeClassifier()
```

```
In [71]: dtree.fit(X_train,y_train)
```

```
Out[71]: DecisionTreeClassifier(ccp_alpha=0.0, class_weight=None, criterion='gini',
                                max_depth=None, max_features=None, max_leaf_nodes=None,
                                min_impurity_decrease=0.0, min_impurity_split=None,
                                min_samples_leaf=1, min_samples_split=2,
                                min_weight_fraction_leaf=0.0, presort='deprecated',
                                random_state=None, splitter='best')
```

فاز هفتم – قسمت اول

Predictions and Evaluation:

اکنون پیش‌بینی‌های مدل را بررسی و ارزیابی می‌کنیم.

```
In [54]: pred = dtree.predict(X_test)
```

```
In [55]: df1_eval=pd.DataFrame({'Label': y_test,'Predicrions':pred})
         df1_eval
```

Out[55]:

	Label	Predicrions
5244	0	0
1739	0	1
2780	0	0
7062	0	1
6661	0	0
...	...	...
9508	0	0
4348	0	1
4233	0	0
5363	0	0
6599	0	0

2874 rows × 2 columns

```
In [56]: print(dtree.score(X_test,y_test))

0.7303409881697982

In [57]: from sklearn.metrics import confusion_matrix
         print(confusion_matrix(y_test,pred))

[[2000  431]
 [ 344   99]]

In [58]: from sklearn.metrics import classification_report
         print(classification_report(y_test,pred))

              precision    recall  f1-score   support

           0       0.85      0.82      0.84      2431
           1       0.19      0.22      0.20       443

    accuracy                           0.73      2874
   macro avg       0.52      0.52      0.52      2874
weighted avg       0.75      0.73      0.74      2874
```

فاز هشتم – قسمت اول

Conclusion:

مشاهده می‌کنیم که score این مدل ، عدد نسبتا خوبی بدست آمده است.

همچنین ماتریس آشفتگی این مدل بیان می‌کند که 2000 نفر از افرادی که پیش بینی شده است وام را پس دهند، آن را برگردانده اند و 344 نفر برنگردانده اند. همچنین 99 نفر از افرادی که پیش بینی شده وام را پس نمی‌دهند آن را بر نگردانده‌اند اما 431 نفر از آنها برگردانده‌اند. تا اینجا به نظر می‌رسد مدل ما کمی بدبین آموزش داده شده است. به همین خاطر به سراغ محاسبه دیگر شاخص ها میرویم و میبینیم شاخص recall برای حالتی که فرد وام را پس نمی‌دهد(1)، 0.22 است، یعنی مدل آموزش داده شده تنها توانسته است 0.22 از افرادی که وام را پس نمی‌دهند، درست شناسایی کند که به معنی است که مدل آموزش داده شده ریسک بسیار بالایی دارد و دقت آن جوابگوی کار ما نیست. البته این مسئله ممکن است با توجه به تعداد کم افرادی که وام را پس نمی‌دهند در مقابل افرادی که وام را پس می‌دهند اتفاق افتاده باشد، یعنی اکثر نمونه‌هایی که مدل با آن آموزش دیده شده، وام را پرداخت کرده بوده‌اند.

فاز ششم – قسمت دوم

Training the Random Forest model:

الگوریتم دیگری که ممکن است در این پیش بینی کمک کننده باشد ، درخت تصادفی است.

برای این منظور الگوریتم خود را به مدل آموزش می‌دهیم و آن را روی داده‌های آموزشی فیت میکنیم.

```
In [79]: from sklearn.ensemble import RandomForestClassifier

In [80]: rfc = RandomForestClassifier(n_estimators=600)

In [81]: rfc.fit(X_train,y_train)

Out[81]: RandomForestClassifier(bootstrap=True, ccp_alpha=0.0, class_weight=None,
                                criterion='gini', max_depth=None, max_features='auto',
                                max_leaf_nodes=None, max_samples=None,
                                min_impurity_decrease=0.0, min_impurity_split=None,
                                min_samples_leaf=1, min_samples_split=2,
                                min_weight_fraction_leaf=0.0, n_estimators=600,
                                n_jobs=None, oob_score=False, random_state=None,
                                verbose=0, warm_start=False)
```

فاز هفتم – قسمت دوم

Predictions and Evaluation

اکنون پیش‌بینی‌های مدل را بررسی و ارزیابی می‌کنیم.

In [64]:
```
rfc_pred = rfc.predict(X_test)
```

In [65]:
```
df_eval=pd.DataFrame({'Label': y_test,'Predicrions':rfc_pred})
df_eval
```

Out[65]:

	Label	Predicrions
5244	0	0
1739	0	0
2780	0	0
7062	0	0
6661	0	0
...	...	...
9508	0	0
4348	0	0
4233	0	0
5363	0	0
6599	0	0

2874 rows × 2 columns

In [66]:
```
print(rfc.score(X_test,y_test))
```

```
0.8444676409185804
```

In [67]:
```
from sklearn.metrics import confusion_matrix
print(confusion_matrix(y_test,rfc_pred))
```

```
[[2420   11]
 [ 436    7]]
```

In [68]:
```
from sklearn.metrics import classification_report
print(classification_report(y_test,rfc_pred))
```

```
              precision    recall  f1-score   support

           0       0.85      1.00      0.92      2431
           1       0.39      0.02      0.03       443

    accuracy                           0.84      2874
   macro avg       0.62      0.51      0.47      2874
weighted avg       0.78      0.84      0.78      2874
```

فاز هشتم – قسمت دوم

Conclusion:

ملاحظه می‌کنیم که دقت مدل افزایش می‌یابد و در این حالت مدل ما recall=1 را برمیگرداند که به این معنی است که مدل توانسته تمام افرادی که وام خود را پس می‌دهند را به درستی پیش بینی کند. و به همان نسبت شاخص recall برای حالتی که فرد وام را پس نمی‌دهد(1)، کاهش یافته است. یعنی از بین افرادی که وام را پس نداده‌اند تنها 0.02 به درستی تشخیص داده اند:

Specificity=TN/N=7/441=0.02