データの読み込み

In [1]:

```
# This Python 3 environment comes with many helpful analytics libraries
 installed
# It is defined by the kaggle/python docker image: https://github.com/ka
ggle/docker-python
# For example, here's several helpful packages to load in

import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)

# Input data files are available in the "../input/" directory.
# For example, running this (by clicking run or pressing Shift+Enter) wi
ll list the files in the input directory

import os
print(os.listdir("../input"))

import seaborn as sns
# Any results you write to the current directory are saved as output.
```

```
['DSS_DMC_Description.pdf', 'carInsurance_train.csv', 'carInsurance_te
st.csv']
```

In [2]:

```
train = pd.read_csv('../input/carInsurance_train.csv')
test = pd.read_csv('../input/carInsurance_test.csv')
```

In [3]:

```
train.head()
```

Out[3]:

| | Id | Age | Job | Marital | Education | Default | Balance | HHInsurance | CarLoan | Comm |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 32 | management | single | tertiary | 0 | 1218 | 1 | 0 | telepho |
| 1 | 2 | 32 | blue-collar | married | primary | 0 | 1156 | 1 | 0 | NaN |
| 2 | 3 | 29 | management | single | tertiary | 0 | 637 | 1 | 0 | cellula |
| 3 | 4 | 25 | student | single | primary | 0 | 373 | 1 | 0 | cellula |
| 4 | 5 | 30 | management | married | tertiary | 0 | 2694 | 0 | 0 | cellula |

In [4]:

```
test.head()
```

Out[4]:

| | Id | Age | Job | Marital | Education | Default | Balance | HHInsurance | CarLoan | Co |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 4001 | 25 | admin. | single | secondary | 0 | 1 | 1 | 1 | Nal |
| 1 | 4002 | 40 | management | married | tertiary | 0 | 0 | 1 | 1 | cell |
| 2 | 4003 | 44 | management | single | tertiary | 0 | -1313 | 1 | 1 | cell |
| 3 | 4004 | 27 | services | single | secondary | 0 | 6279 | 1 | 0 | cell |
| 4 | 4005 | 53 | technician | married | secondary | 0 | 7984 | 1 | 0 | cell |

データの確認

In [5]:
```
train.isna().sum()
```

Out[5]:
```
Id                     0
Age                    0
Job                   19
Marital                0
Education            169
Default                0
Balance                0
HHInsurance            0
CarLoan                0
Communication        902
LastContactDay         0
LastContactMonth       0
NoOfContacts           0
DaysPassed             0
PrevAttempts           0
Outcome             3042
CallStart              0
CallEnd                0
CarInsurance           0
dtype: int64
```

In [6]:
```
test.isna().sum()
```

Out[6]:
```
Id                     0
Age                    0
Job                    5
Marital                0
Education             47
Default                0
Balance                0
HHInsurance            0
CarLoan                0
Communication        221
LastContactDay         0
LastContactMonth       0
NoOfContacts           0
DaysPassed             0
PrevAttempts           0
Outcome              757
CallStart              0
CallEnd                0
CarInsurance        1000
dtype: int64
```

In [7]:
```
train.shape
```

Out[7]:

```
(4000, 19)
```

In [8]:
```
test.shape
```

Out[8]:
```
(1000, 19)
```

## 欠損値の対応

In [9]:
```
train.Job.unique()
```

Out[9]:
```
array(['management', 'blue-collar', 'student', 'technician', 'admin.',
       'services', 'self-employed', 'retired', nan, 'housemaid',
       'entrepreneur', 'unemployed'], dtype=object)
```

In [10]:
```
train[train.Job != train.Job].head()
```

Out[10]:

| | Id | Age | Job | Marital | Education | Default | Balance | HHInsurance | CarLoan | Commun |
|---|---|---|---|---|---|---|---|---|---|---|
| 27 | 28 | 45 | NaN | divorced | NaN | 0 | 0 | 0 | 0 | cellular |
| 239 | 240 | 41 | NaN | single | NaN | 0 | 942 | 0 | 0 | cellular |
| 486 | 487 | 54 | NaN | married | primary | 0 | 981 | 0 | 0 | cellular |
| 536 | 537 | 33 | NaN | single | secondary | 0 | 1522 | 0 | 1 | cellular |
| 605 | 606 | 53 | NaN | married | primary | 0 | 732 | 0 | 0 | cellular |

In [11]:
```
train.Job.fillna('unknown', inplace=True)
test.Job.fillna('unknown', inplace=True)
```

In [12]:
```
train.Education.unique()
```

Out[12]:
```
array(['tertiary', 'primary', 'secondary', nan], dtype=object)
```

In [13]:
```
train.Education.fillna('other', inplace=True)
test.Education.fillna('other', inplace=True)
```

In [14]:
```
train.Communication.fillna('other', inplace=True)
test.Communication.fillna('other', inplace=True)
```

In [15]:
```
train.Outcome.fillna('unknwon', inplace=True)
test.Outcome.fillna('unknwon', inplace=True)
```

In [16]:
```
train.dtypes
```

```
train.dtypes
```

Out[16]:

```
Id                   int64
Age                  int64
Job                 object
Marital             object
Education           object
Default              int64
Balance              int64
HHInsurance          int64
CarLoan              int64
Communication       object
LastContactDay       int64
LastContactMonth    object
NoOfContacts         int64
DaysPassed           int64
PrevAttempts         int64
Outcome             object
CallStart           object
CallEnd             object
CarInsurance         int64
dtype: object
```

In [17]:

```
cat_cols = ['Marital', 'Education', 'Job', 'Communication',
            'LastContactMonth', 'Outcome', 'CallStart', 'CallEnd']
```

In [18]:

```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
for col in cat_cols:
    train[col] = le.fit_transform(train[col])
```

In [19]:

```
train.head()
```

Out[19]:

| | Id | Age | Job | Marital | Education | Default | Balance | HHInsurance | CarLoan | Communication |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 32 | 4 | 2 | 3 | 0 | 1218 | 1 | 0 | 2 |
| 1 | 2 | 32 | 1 | 1 | 1 | 0 | 1156 | 1 | 0 | 1 |
| 2 | 3 | 29 | 4 | 2 | 3 | 0 | 637 | 1 | 0 | 0 |
| 3 | 4 | 25 | 8 | 2 | 1 | 0 | 373 | 1 | 0 | 0 |
| 4 | 5 | 30 | 4 | 1 | 3 | 0 | 2694 | 0 | 0 | 0 |

In [20]:

```
X = train.drop('CarInsurance', axis=1)
y = train.CarInsurance
```

In [21]:

```
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)
```

In [22]:

```
import xgboost as xgb
dtrain = xgb.DMatrix(X_train, y_train)
dval = xgb.DMatrix(X_test, y_test)
```

```
eval_list = [(dval, 'eval'), (dtrain, 'train')]
```

In [23]:
```
param = {'max_depth': 3,
         'eta': 0.3,
         'silent': 1,
         'objective': 'binary:logistic',
         'nthread' : 4,
         'eval_metric' : 'auc'}
```

In [24]:
```
num_round = 250
bst = xgb.train(param, dtrain, num_round, eval_list, early_stopping_ro
unds=2)
```

```
[0]	eval-auc:0.695181	train-auc:0.697895
Multiple eval metrics have been passed: 'train-auc' will be used for e
arly stopping.

Will train until train-auc hasn't improved in 2 rounds.
[1]	eval-auc:0.716765	train-auc:0.717916
[2]	eval-auc:0.719608	train-auc:0.720543
[3]	eval-auc:0.729629	train-auc:0.733879
[4]	eval-auc:0.736563	train-auc:0.749381
[5]	eval-auc:0.743144	train-auc:0.754882
[6]	eval-auc:0.749341	train-auc:0.758067
[7]	eval-auc:0.748643	train-auc:0.768482
[8]	eval-auc:0.744592	train-auc:0.773661
[9]	eval-auc:0.746565	train-auc:0.78172
[10]	eval-auc:0.752592	train-auc:0.787233
[11]	eval-auc:0.754994	train-auc:0.788592
[12]	eval-auc:0.756681	train-auc:0.792386
[13]	eval-auc:0.759272	train-auc:0.804491
[14]	eval-auc:0.760326	train-auc:0.809091
[15]	eval-auc:0.762587	train-auc:0.813732
[16]	eval-auc:0.762656	train-auc:0.81744
[17]	eval-auc:0.760549	train-auc:0.820491
[18]	eval-auc:0.760891	train-auc:0.822419
[19]	eval-auc:0.762532	train-auc:0.825748
[20]	eval-auc:0.763818	train-auc:0.828178
[21]	eval-auc:0.765721	train-auc:0.835211
[22]	eval-auc:0.768948	train-auc:0.83791
[23]	eval-auc:0.770859	train-auc:0.841895
[24]	eval-auc:0.771118	train-auc:0.843803
[25]	eval-auc:0.770528	train-auc:0.84554
[26]	eval-auc:0.774272	train-auc:0.847045
[27]	eval-auc:0.775409	train-auc:0.847526
[28]	eval-auc:0.774368	train-auc:0.851024
[29]	eval-auc:0.776482	train-auc:0.854434
[30]	eval-auc:0.777776	train-auc:0.855469
[31]	eval-auc:0.774903	train-auc:0.857863
[32]	eval-auc:0.777768	train-auc:0.862475
[33]	eval-auc:0.78054	train-auc:0.868367
[34]	eval-auc:0.779721	train-auc:0.869234
[35]	eval-auc:0.780736	train-auc:0.870733
[36]	eval-auc:0.780802	train-auc:0.871111
[37]	eval-auc:0.780696	train-auc:0.871691
[38]	eval-auc:0.781193	train-auc:0.872157
[39]	eval-auc:0.785282	train-auc:0.878938
[40]	eval-auc:0.785719	train-auc:0.879086
[41]	eval-auc:0.788003	train-auc:0.8816
[42]	eval-auc:0.789107	train-auc:0.884406
```

```
[43]	eval-auc:0.792159	train-auc:0.886732
[44]	eval-auc:0.793712	train-auc:0.887952
[45]	eval-auc:0.793683	train-auc:0.89024
[46]	eval-auc:0.793962	train-auc:0.890456
[47]	eval-auc:0.795044	train-auc:0.89139
[48]	eval-auc:0.795015	train-auc:0.892265
[49]	eval-auc:0.795886	train-auc:0.893102
[50]	eval-auc:0.795012	train-auc:0.894438
[51]	eval-auc:0.793657	train-auc:0.89577
[52]	eval-auc:0.792455	train-auc:0.896475
[53]	eval-auc:0.792009	train-auc:0.89688
[54]	eval-auc:0.794986	train-auc:0.898259
[55]	eval-auc:0.798415	train-auc:0.901852
[56]	eval-auc:0.799062	train-auc:0.902795
[57]	eval-auc:0.799223	train-auc:0.905375
[58]	eval-auc:0.798961	train-auc:0.906283
[59]	eval-auc:0.799082	train-auc:0.906397
[60]	eval-auc:0.799968	train-auc:0.908464
[61]	eval-auc:0.804492	train-auc:0.912678
[62]	eval-auc:0.805033	train-auc:0.913046
[63]	eval-auc:0.805246	train-auc:0.913231
[64]	eval-auc:0.807371	train-auc:0.915818
[65]	eval-auc:0.80734	train-auc:0.917287
[66]	eval-auc:0.808812	train-auc:0.918779
[67]	eval-auc:0.807915	train-auc:0.91947
[68]	eval-auc:0.808027	train-auc:0.920107
[69]	eval-auc:0.808332	train-auc:0.921876
[70]	eval-auc:0.807072	train-auc:0.922467
[71]	eval-auc:0.806802	train-auc:0.922621
[72]	eval-auc:0.807222	train-auc:0.924367
[73]	eval-auc:0.806983	train-auc:0.924595
[74]	eval-auc:0.807112	train-auc:0.926076
[75]	eval-auc:0.809157	train-auc:0.926806
[76]	eval-auc:0.808559	train-auc:0.928867
[77]	eval-auc:0.808254	train-auc:0.929176
[78]	eval-auc:0.80748	train-auc:0.930279
[79]	eval-auc:0.808303	train-auc:0.930843
[80]	eval-auc:0.807938	train-auc:0.931084
[81]	eval-auc:0.813164	train-auc:0.933673
[82]	eval-auc:0.813592	train-auc:0.934851
[83]	eval-auc:0.814191	train-auc:0.934973
[84]	eval-auc:0.814944	train-auc:0.936082
[85]	eval-auc:0.815485	train-auc:0.937128
[86]	eval-auc:0.815414	train-auc:0.937518
[87]	eval-auc:0.815604	train-auc:0.937653
[88]	eval-auc:0.815128	train-auc:0.939501
[89]	eval-auc:0.814179	train-auc:0.941096
[90]	eval-auc:0.815094	train-auc:0.941662
[91]	eval-auc:0.815706	train-auc:0.94207
[92]	eval-auc:0.81715	train-auc:0.944309
[93]	eval-auc:0.822414	train-auc:0.947442
[94]	eval-auc:0.823559	train-auc:0.94873
[95]	eval-auc:0.822923	train-auc:0.949173
[96]	eval-auc:0.824591	train-auc:0.951035
[97]	eval-auc:0.824134	train-auc:0.951356
[98]	eval-auc:0.824418	train-auc:0.951512
[99]	eval-auc:0.824531	train-auc:0.951993
[100]	eval-auc:0.824367	train-auc:0.952391
[101]	eval-auc:0.822969	train-auc:0.953228
[102]	eval-auc:0.821715	train-auc:0.953477
[103]	eval-auc:0.822931	train-auc:0.954154
[104]	eval-auc:0.823182	train-auc:0.954344
[105]	eval-auc:0.822693	train-auc:0.955018
[106]	eval-auc:0.826035	train-auc:0.956926
```

```
[107]	eval-auc:0.826302	train-auc:0.956933
[108]	eval-auc:0.827418	train-auc:0.958314
[109]	eval-auc:0.827473	train-auc:0.959437
[110]	eval-auc:0.828005	train-auc:0.960222
[111]	eval-auc:0.82856	train-auc:0.960485
[112]	eval-auc:0.828721	train-auc:0.960666
[113]	eval-auc:0.829561	train-auc:0.961082
[114]	eval-auc:0.830293	train-auc:0.962139
[115]	eval-auc:0.830438	train-auc:0.962489
[116]	eval-auc:0.831514	train-auc:0.96279
[117]	eval-auc:0.832305	train-auc:0.963713
[118]	eval-auc:0.833225	train-auc:0.964755
[119]	eval-auc:0.833858	train-auc:0.965132
[120]	eval-auc:0.834232	train-auc:0.965139
[121]	eval-auc:0.83456	train-auc:0.965365
[122]	eval-auc:0.833666	train-auc:0.965891
[123]	eval-auc:0.834244	train-auc:0.966339
[124]	eval-auc:0.835176	train-auc:0.966881
[125]	eval-auc:0.835026	train-auc:0.966951
[126]	eval-auc:0.835035	train-auc:0.967118
[127]	eval-auc:0.834813	train-auc:0.968371
[128]	eval-auc:0.832886	train-auc:0.969248
[129]	eval-auc:0.832774	train-auc:0.969466
[130]	eval-auc:0.836657	train-auc:0.971034
[131]	eval-auc:0.836763	train-auc:0.971208
[132]	eval-auc:0.835204	train-auc:0.97146
[133]	eval-auc:0.83527	train-auc:0.972203
[134]	eval-auc:0.836082	train-auc:0.972473
[135]	eval-auc:0.836211	train-auc:0.972891
[136]	eval-auc:0.837445	train-auc:0.973243
[137]	eval-auc:0.838256	train-auc:0.973903
[138]	eval-auc:0.838662	train-auc:0.974633
[139]	eval-auc:0.841313	train-auc:0.975875
[140]	eval-auc:0.83997	train-auc:0.9762
[141]	eval-auc:0.840819	train-auc:0.976585
[142]	eval-auc:0.842403	train-auc:0.977549
[143]	eval-auc:0.840781	train-auc:0.977851
[144]	eval-auc:0.841748	train-auc:0.978645
[145]	eval-auc:0.841437	train-auc:0.979061
[146]	eval-auc:0.842389	train-auc:0.979404
[147]	eval-auc:0.842752	train-auc:0.979594
[148]	eval-auc:0.842872	train-auc:0.980057
[149]	eval-auc:0.843169	train-auc:0.98027
[150]	eval-auc:0.843801	train-auc:0.980336
[151]	eval-auc:0.84379	train-auc:0.980378
[152]	eval-auc:0.843194	train-auc:0.980438
[153]	eval-auc:0.842775	train-auc:0.980534
[154]	eval-auc:0.842933	train-auc:0.980654
[155]	eval-auc:0.842596	train-auc:0.980933
[156]	eval-auc:0.842268	train-auc:0.980957
[157]	eval-auc:0.842383	train-auc:0.981218
[158]	eval-auc:0.843968	train-auc:0.981196
[159]	eval-auc:0.842987	train-auc:0.981407
[160]	eval-auc:0.843197	train-auc:0.981558
[161]	eval-auc:0.843171	train-auc:0.981657
[162]	eval-auc:0.842619	train-auc:0.981642
[163]	eval-auc:0.842438	train-auc:0.981931
[164]	eval-auc:0.841595	train-auc:0.982045
[165]	eval-auc:0.841736	train-auc:0.982354
[166]	eval-auc:0.84182	train-auc:0.982449
[167]	eval-auc:0.841679	train-auc:0.982654
[168]	eval-auc:0.84295	train-auc:0.983102
[169]	eval-auc:0.842907	train-auc:0.983235
[170]	eval-auc:0.843876	train-auc:0.983465
```

```
[171]	eval-auc:0.844932	train-auc:0.984307
[172]	eval-auc:0.845432	train-auc:0.98495
[173]	eval-auc:0.845774	train-auc:0.985269
[174]	eval-auc:0.845283	train-auc:0.98555
[175]	eval-auc:0.845582	train-auc:0.985811
[176]	eval-auc:0.845271	train-auc:0.985785
[177]	eval-auc:0.846853	train-auc:0.986212
[178]	eval-auc:0.846347	train-auc:0.986379
[179]	eval-auc:0.846281	train-auc:0.986476
[180]	eval-auc:0.844365	train-auc:0.986599
[181]	eval-auc:0.845234	train-auc:0.986931
[182]	eval-auc:0.845055	train-auc:0.98702
[183]	eval-auc:0.845271	train-auc:0.98712
[184]	eval-auc:0.845613	train-auc:0.987229
[185]	eval-auc:0.845461	train-auc:0.98731
[186]	eval-auc:0.845866	train-auc:0.987417
[187]	eval-auc:0.846298	train-auc:0.987602
[188]	eval-auc:0.846928	train-auc:0.987732
[189]	eval-auc:0.846801	train-auc:0.987731
[190]	eval-auc:0.84666	train-auc:0.987911
[191]	eval-auc:0.84639	train-auc:0.987988
[192]	eval-auc:0.845947	train-auc:0.988237
[193]	eval-auc:0.845726	train-auc:0.988308
[194]	eval-auc:0.84559	train-auc:0.988511
[195]	eval-auc:0.845636	train-auc:0.988753
[196]	eval-auc:0.844779	train-auc:0.988883
[197]	eval-auc:0.844584	train-auc:0.988891
[198]	eval-auc:0.844425	train-auc:0.989016
[199]	eval-auc:0.844995	train-auc:0.989119
[200]	eval-auc:0.844811	train-auc:0.989369
[201]	eval-auc:0.845113	train-auc:0.98974
[202]	eval-auc:0.846203	train-auc:0.990066
[203]	eval-auc:0.84574	train-auc:0.990109
[204]	eval-auc:0.846902	train-auc:0.990421
[205]	eval-auc:0.847509	train-auc:0.990742
[206]	eval-auc:0.848173	train-auc:0.990919
[207]	eval-auc:0.84851	train-auc:0.990924
[208]	eval-auc:0.848976	train-auc:0.991109
[209]	eval-auc:0.849301	train-auc:0.991292
[210]	eval-auc:0.849818	train-auc:0.991461
[211]	eval-auc:0.84922	train-auc:0.991518
[212]	eval-auc:0.848861	train-auc:0.991459
[213]	eval-auc:0.84897	train-auc:0.991507
Stopping. Best iteration:
[211]	eval-auc:0.84922	train-auc:0.991518
```

## 特徴エンジニアリング

In [25]:

```
def read_and_fillna():
    train = pd.read_csv('../input/carInsurance_train.csv')
    test = pd.read_csv('../input/carInsurance_test.csv')
    train.Job.fillna('unknown', inplace=True)
    test.Job.fillna('unknown', inplace=True)
    train.Education.fillna('other', inplace=True)
    test.Education.fillna('other', inplace=True)
    train.Communication.fillna('other', inplace=True)
    test.Communication.fillna('other', inplace=True)
```

```
    train.Outcome.fillna('unknwon', inplace=True)
    test.Outcome.fillna('unknwon', inplace=True)
    return train, test
```

In [26]:

```
def make_model(train, cat_cols):
    from sklearn.preprocessing import LabelEncoder
    from sklearn.model_selection import train_test_split
    import xgboost as xgb

    le = LabelEncoder()
    for col in cat_cols:
        train[col] = le.fit_transform(train[col])
    X = train.drop('CarInsurance', axis=1)
    y = train.CarInsurance
    X_train, X_test, y_train, y_test = train_test_split(X, y,
                                                         test_size=0.3,
random_state=0)
    dtrain = xgb.DMatrix(X_train, y_train)
    dval = xgb.DMatrix(X_test, y_test)
    eval_list = [(dval, 'eval'), (dtrain, 'train')]
    param = {'max_depth': 3,
             'eta': 0.3,
             'silent': 1,
             'objective': 'binary:logistic',
             'nthread' : 4,
             'eval_metric' : 'auc'}

    num_round = 250
    bst = xgb.train(param, dtrain, num_round, eval_list, early_stoppin
g_rounds=2)
```

In [27]:

```
train, test = read_and_fillna()
```

## Age

In [28]:

```
train.Age.describe()
```

Out[28]:

```
count    4000.000000
mean       41.214750
std        11.550194
min        18.000000
25%        32.000000
50%        39.000000
75%        49.000000
max        95.000000
Name: Age, dtype: float64
```

In [29]:

```
def make_AgeGroup(df):
    df["AgeGroup"] = df.Age.apply(lambda x : 0 if x >= 10 and x <= 19
else
                                    (1 if x >= 20 and x <= 29 else
                                     (2 if x >= 30 and x <= 39 else
                                      (3 if x >= 40 and x <= 49 else
                                       (4 if x >= 50 and x <= 59 else
                                        (5 if x >= 60 and x <= 69 else
```

```
                                        (6 if x >= 70 and x <= 79 els
e
                                         (7 if x >= 80 and x <= 80 el
se 8))))))))
```

In [30]:

```
make_AgeGroup(train)
make_AgeGroup(test)
```

In [31]:

```
pd.pivot_table(train[['AgeGroup', 'CarInsurance', 'Age']], columns=['C
arInsurance'], index=['AgeGroup'], aggfunc=['count'])
```

Out[31]:

| | count | |
|---|---|---|
| | Age | |
| CarInsurance | 0 | 1 |
| AgeGroup | | |
| 0 | 4 | 11 |
| 1 | 257 | 261 |
| 2 | 980 | 558 |
| 3 | 635 | 353 |
| 4 | 454 | 247 |
| 5 | 44 | 114 |
| 6 | 17 | 43 |
| 7 | 2 | 7 |
| 8 | 3 | 10 |

## Job

In [32]:

```
train.Job.unique()
```

Out[32]:

```
array(['management', 'blue-collar', 'student', 'technician', 'admin.',
       'services', 'self-employed', 'retired', 'unknown', 'housemaid',
       'entrepreneur', 'unemployed'], dtype=object)
```

In [33]:

```
pd.pivot_table(train[['Job', 'CarInsurance', 'Age']], columns=['CarIns
urance'], index=['Job'], aggfunc=['count'])
```

Out[33]:

| | count | |
|---|---|---|
| | Age | |
| CarInsurance | 0 | 1 |
| Job | | |
| admin. | 274 | 185 |
| blue-collar | 540 | 219 |
| entrepreneur | 86 | 35 |
| housemaid | 72 | 37 |
| management | 501 | 392 |

| | | |
|---|---|---|
| retired | 103 | 146 |
| self-employed | 86 | 54 |
| services | 218 | 112 |
| student | 44 | 87 |
| technician | 406 | 254 |



In [34]:
```
def make_isWork(df):
    df['isWork'] = df.Job.apply(lambda x : 0 if x == "retired"
                                or x == "student"
                                or x == "unemployed"
                                else 1)
```

In [35]:
```
make_isWork(train)
make_isWork(test)
```

In [36]:
```
train.drop('Job', axis=1, inplace=True)
test.drop('Job', axis=1, inplace=True)
```

## Martial

In [37]:
```
pd.pivot_table(train[['Marital', 'CarInsurance', 'Age']], columns=['Ca
rInsurance'], index=['Marital'], aggfunc=['count'])
```

Out[37]:

| | count | |
|---|---|---|
| | Age | |
| CarInsurance | 0 | 1 |
| Marital | | |
| divorced | 273 | 210 |
| married | 1471 | 833 |
| single | 652 | 561 |

## Education

In [38]:
```
pd.pivot_table(train[['Education', 'CarInsurance', 'Age']], columns=[
'CarInsurance'], index=['Education'], aggfunc=['count'])
```

Out[38]:

| | count | |
|---|---|---|
| | Age | |
| CarInsurance | 0 | 1 |

| Education | | |
|---|---|---|
| other | 90 | 79 |
| primary | 366 | 195 |
| secondary | 1258 | 730 |
| tertiary | 682 | 600 |

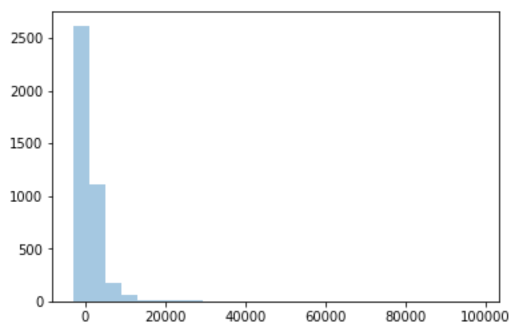


## Balance

In [39]:
```
train.Balance = train.Balance.astype('float')
test.Balance = test.Balance.astype('float')
```

In [40]:
```
sns.distplot(train.Balance.values, kde=False, rug=False, bins=25)
```

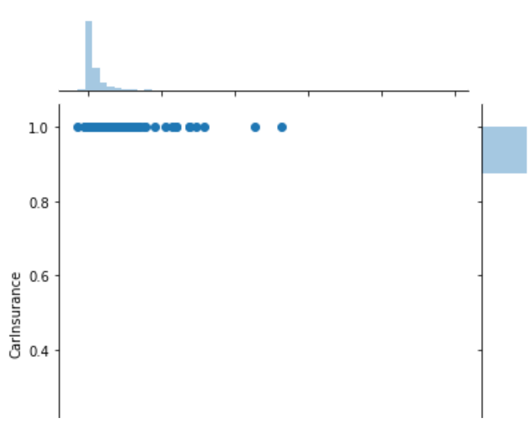
Out[40]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7ff4daa4b7b8>
```

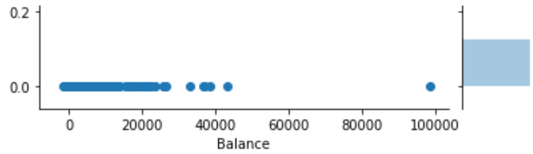



In [41]:
```
sns.jointplot('Balance', 'CarInsurance', data=train)
```

```
/opt/conda/lib/python3.6/site-packages/scipy/stats/stats.py:1713: Futu
reWarning: Using a non-tuple sequence for multidimensional indexing is
deprecated; use `arr[tuple(seq)]` instead of `arr[seq]`. In the future
this will be interpreted as an array index, `arr[np.array(seq)]`, whic
h will result either in an error or a different result.
  return np.add.reduce(sorted[indexer] * weights, axis=axis) / sumval
```

Out[41]:
```
<seaborn.axisgrid.JointGrid at 0x7ff4d918acf8>
```

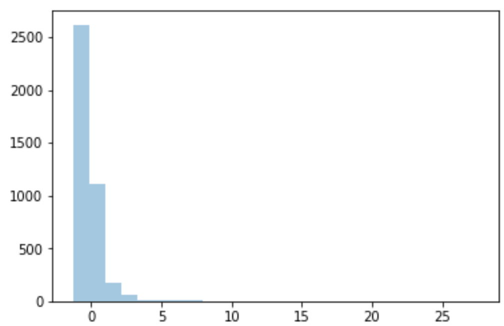





In [42]:
```
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
train.Balance = sc.fit_transform(train.Balance.values.reshape(-1, 1))
test.Balance = sc.fit_transform(test.Balance.values.reshape(-1, 1))
```

In [43]:
```
sns.distplot(train.Balance.values, kde=False, rug=False, bins=25)
```

Out[43]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7ff4d8775cc0>
```




In [44]:
```
train.Balance.describe()
```

Out[44]:
```
count    4.000000e+03
mean     4.639344e-17
std      1.000125e+00
min     -1.307582e+00
25%     -4.049934e-01
50%     -2.795310e-01
75%      2.451222e-02
max      2.759433e+01
Name: Balance, dtype: float64
```

## Communication

In [45]:
```
train.Communication.unique()
```

Out[45]:
```
array(['telephone', 'other', 'cellular'], dtype=object)
```

In [46]:
```
pd.pivot_table(train[['Communication', 'CarInsurance', 'Age']], column
s=['CarInsurance'], index=['Communication'], aggfunc=['count'])
```

Out[46]:

| | count |
|---|---|

| | Age | |
|---|---|---|
| CarInsurance | 0 | 1 |
| Communication | | |
| cellular | 1518 | 1313 |
| other | 734 | 168 |
| telephone | 144 | 123 |

In [47]:

```
train["isMobile"] = train.Communication.apply(lambda x : 1 if x == "Ce
llular" or x == "telephone" else 0)
test["isMobile"] = test.Communication.apply(lambda x : 1 if x == "Cell
ular" or x == "telephone" else 0)
```

## LastContactDay & Month

In [48]:

```
pd.pivot_table(train[['LastContactMonth', 'CarInsurance', 'Age']], col
umns=['CarInsurance'], index=['LastContactMonth'], aggfunc=['count'])
```

Out[48]:

| | count | |
|---|---|---|
| | Age | |
| CarInsurance | 0 | 1 |
| LastContactMonth | | |
| apr | 150 | 156 |
| aug | 342 | 194 |
| dec | 7 | 34 |
| feb | 129 | 133 |
| jan | 86 | 48 |
| jul | 364 | 209 |
| jun | 283 | 171 |
| mar | 15 | 64 |
| may | 760 | 289 |
| nov | 215 | 132 |
| oct | 27 | 91 |
| sep | 18 | 83 |

## NoOfContacts

In [49]:

```
pd.pivot_table(train[['NoOfContacts', 'CarInsurance', 'Age']], columns
=['CarInsurance'], index=['NoOfContacts'], aggfunc=['count'])
```

Out[49]:

| | count | |
|---|---|---|
| | Age | |
| CarInsurance | 0 | 1 |
| NoOfContacts | | |

| NoOfContacts | | |
|---|---|---|
| 1 | 912.0 | 773.0 |
| 2 | 671.0 | 414.0 |
| 3 | 311.0 | 205.0 |
| 4 | 156.0 | 81.0 |
| 5 | 114.0 | 52.0 |
| 6 | 62.0 | 26.0 |
| 7 | 34.0 | 15.0 |
| 8 | 27.0 | 14.0 |
| 9 | 18.0 | 2.0 |
| 10 | 13.0 | 5.0 |
| 11 | 7.0 | 8.0 |
| 12 | 11.0 | NaN |
| 13 | 6.0 | 2.0 |
| 14 | 6.0 | 1.0 |
| 15 | 2.0 | 1.0 |
| 16 | 3.0 | NaN |
| 17 | 9.0 | 2.0 |
| 18 | 3.0 | NaN |
| 19 | 3.0 | NaN |
| 20 | 4.0 | NaN |
| 21 | 3.0 | 1.0 |
| 22 | 3.0 | NaN |
| 23 | 3.0 | NaN |
| 24 | 2.0 | 1.0 |
| 25 | 4.0 | NaN |
| 26 | 1.0 | NaN |
| 27 | 1.0 | NaN |
| 28 | 1.0 | NaN |
| 29 | NaN | 1.0 |
| 30 | 1.0 | NaN |
| 32 | 1.0 | NaN |
| 34 | 1.0 | NaN |
| 38 | 1.0 | NaN |
| 41 | 1.0 | NaN |
| 43 | 1.0 | NaN |

## DaysPassed

In [50]:

```
train.DaysPassed.value_counts()
```

Out[50]:

```
-1      3042
 92       38
 182      33
 91       24
 183      24
 93       16
 95       16
 94       14
```

```
97        13
181       13
189       12
184       12
90        11
178       10
370       10
105        9
196        9
104        8
350        8
185        8
188        8
98         8
169        8
195        7
187        7
176        6
175        6
88         6
168        6
343        6
          ...
310        1
308        1
474        1
532        1
127        1
544        1
121        1
115        1
73         1
71         1
67         1
65         1
61         1
57         1
53         1
49         1
43         1
37         1
35         1
27         1
21         1
15         1
13         1
854        1
775        1
828        1
728        1
690        1
558        1
842        1
Name: DaysPassed, Length: 330, dtype: int64
```

In [51]:

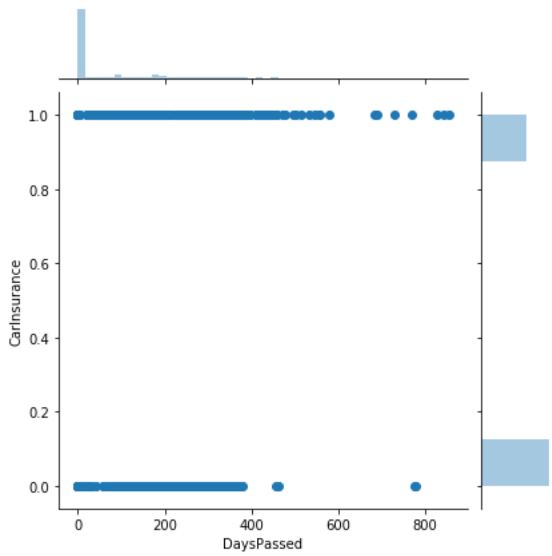
```
sns.jointplot('DaysPassed', 'CarInsurance', data=train)
```

```
/opt/conda/lib/python3.6/site-packages/scipy/stats/stats.py:1713: Futu
reWarning: Using a non-tuple sequence for multidimensional indexing is
deprecated; use `arr[tuple(seq)]` instead of `arr[seq]`. In the future
this will be interpreted as an array index, `arr[np.array(seq)]`, whic
h will result either in an error or a different result.
```

```
    return np.add.reduce(sorted[indexer] * weights, axis=axis) / sumval
```

Out[51]:

```
<seaborn.axisgrid.JointGrid at 0x7ff4d8731320>
```




## PrevAttempts

In [52]:

```
train.PrevAttempts.value_counts()
```

Out[52]:

```
0     3042
1      335
2      251
3      125
4       79
5       60
6       25
7       21
8       18
10      10
9        9
14       5
12       5
13       4
19       4
11       3
23       1
18       1
58       1
30       1
Name: PrevAttempts, dtype: int64
```

In [53]:

```
pd.pivot_table(train[['PrevAttempts', 'CarInsurance', 'Age']], columns
=['CarInsurance'], index=['PrevAttempts'], aggfunc=['count'])
```

Out[53]:

| | count |
|---|---|
| | Age |

| CarInsurance | 0 | 1 |
|---|---|---|
| PrevAttempts | | |
| 0 | 1997.0 | 1045.0 |
| 1 | 149.0 | 186.0 |
| 2 | 109.0 | 142.0 |
| 3 | 47.0 | 78.0 |
| 4 | 33.0 | 46.0 |
| 5 | 26.0 | 34.0 |
| 6 | 9.0 | 16.0 |
| 7 | 6.0 | 15.0 |
| 8 | 4.0 | 14.0 |
| 9 | 5.0 | 4.0 |
| 10 | NaN | 10.0 |
| 11 | 1.0 | 2.0 |
| 12 | 1.0 | 4.0 |
| 13 | 2.0 | 2.0 |
| 14 | 3.0 | 2.0 |
| 18 | 1.0 | NaN |
| 19 | 2.0 | 2.0 |
| 23 | 1.0 | NaN |
| 30 | NaN | 1.0 |
| 58 | NaN | 1.0 |

## Outcom

In [54]:

```
train.Outcome.unique()
```

Out[54]:

```
array(['unknwon', 'failure', 'other', 'success'], dtype=object)
```

In [55]:

```
pd.pivot_table(train[['Outcome', 'CarInsurance', 'Age']], columns=['Ca
rInsurance'], index=['Outcome'], aggfunc=['count'])
```

Out[55]:

| | count | |
|---|---|---|
| | Age | |
| CarInsurance | 0 | 1 |
| Outcome | | |
| failure | 261 | 176 |
| other | 103 | 92 |
| success | 35 | 291 |
| unknwon | 1997 | 1045 |

## Time

In [56]:

```
train['isCallStart'] = 0
train['isCallEnd'] = 0
test['isCallStart'] = 0
test['isCallEnd'] = 0

for idx, time in enumerate(train.CallStart):
    (h, m, s) = time.split(':')
    result = int(h) * 3600 + int(m) * 60 + int(s)
    train.loc[idx, 'isCallStart'] = result

for idx, time in enumerate(train.CallEnd):
    (h, m, s) = time.split(':')
    result = int(h) * 3600 + int(m) * 60 + int(s)
    train.loc[idx, 'isCallEnd'] = result
```

In [57]:

```
def make_time_column(df):
    df["diffTime"] = df.isCallStart - df.isCallEnd
    df[['isCallStart', 'isCallEnd', 'diffTime']] = df[['isCallStart',
'isCallEnd', 'diffTime']].astype("float")
    df.drop(['CallStart', 'CallEnd'], axis=1, inplace=True)
```

In [58]:

```
make_time_column(train)
make_time_column(test)
```

In [59]:

```
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
train[['isCallStart', 'isCallEnd', 'diffTime']] = sc.fit_transform(tra
in[['isCallStart', 'isCallEnd', 'diffTime']])
test[['isCallStart', 'isCallEnd', 'diffTime']] = sc.fit_transform(test
[['isCallStart', 'isCallEnd', 'diffTime']])
```

In [60]:

```
cat_cols = ['Marital', 'Education', 'Communication',
            'LastContactMonth', 'Outcome']

make_model(train, cat_cols)
```

```
[0]	eval-auc:0.805461	train-auc:0.843845
Multiple eval metrics have been passed: 'train-auc' will be used for e
arly stopping.

Will train until train-auc hasn't improved in 2 rounds.
[1]	eval-auc:0.854935	train-auc:0.878551
[2]	eval-auc:0.861044	train-auc:0.887408
[3]	eval-auc:0.862193	train-auc:0.88986
[4]	eval-auc:0.86773	train-auc:0.898391
[5]	eval-auc:0.873595	train-auc:0.903397
[6]	eval-auc:0.875166	train-auc:0.905741
[7]	eval-auc:0.879566	train-auc:0.909439
[8]	eval-auc:0.885511	train-auc:0.914387
[9]	eval-auc:0.886972	train-auc:0.917139
[10]	eval-auc:0.88952	train-auc:0.918847
[11]	eval-auc:0.890745	train-auc:0.921649
[12]	eval-auc:0.892542	train-auc:0.923362
[13]	eval-auc:0.893384	train-auc:0.924535
[14]	eval-auc:0.895435	train-auc:0.927988
[15]	eval-auc:0.897425	train-auc:0.929414
```

```
[16]	eval-auc:0.897631	train-auc:0.930781
[17]	eval-auc:0.897595	train-auc:0.932208
[18]	eval-auc:0.89748	train-auc:0.93341
[19]	eval-auc:0.898585	train-auc:0.934499
[20]	eval-auc:0.901816	train-auc:0.937007
[21]	eval-auc:0.902859	train-auc:0.938091
[22]	eval-auc:0.90287	train-auc:0.939094
[23]	eval-auc:0.904182	train-auc:0.941049
[24]	eval-auc:0.904511	train-auc:0.942365
[25]	eval-auc:0.905828	train-auc:0.943278
[26]	eval-auc:0.905926	train-auc:0.94393
[27]	eval-auc:0.905302	train-auc:0.945311
[28]	eval-auc:0.905516	train-auc:0.945758
[29]	eval-auc:0.905237	train-auc:0.946369
[30]	eval-auc:0.905542	train-auc:0.946539
[31]	eval-auc:0.905686	train-auc:0.947442
[32]	eval-auc:0.905301	train-auc:0.948264
[33]	eval-auc:0.905076	train-auc:0.949809
[34]	eval-auc:0.906362	train-auc:0.950933
[35]	eval-auc:0.906819	train-auc:0.951552
[36]	eval-auc:0.906471	train-auc:0.952219
[37]	eval-auc:0.906581	train-auc:0.953686
[38]	eval-auc:0.906885	train-auc:0.954887
[39]	eval-auc:0.90738	train-auc:0.955329
[40]	eval-auc:0.907521	train-auc:0.955581
[41]	eval-auc:0.909183	train-auc:0.95686
[42]	eval-auc:0.909503	train-auc:0.957208
[43]	eval-auc:0.909126	train-auc:0.958158
[44]	eval-auc:0.909629	train-auc:0.959115
[45]	eval-auc:0.90977	train-auc:0.959549
[46]	eval-auc:0.910553	train-auc:0.960308
[47]	eval-auc:0.911131	train-auc:0.961035
[48]	eval-auc:0.911602	train-auc:0.961608
[49]	eval-auc:0.911832	train-auc:0.962787
[50]	eval-auc:0.911404	train-auc:0.963231
[51]	eval-auc:0.9108 train-auc:0.96373
[52]	eval-auc:0.910699	train-auc:0.964128
[53]	eval-auc:0.910403	train-auc:0.965209
[54]	eval-auc:0.910676	train-auc:0.965479
[55]	eval-auc:0.910578	train-auc:0.966197
[56]	eval-auc:0.910351	train-auc:0.966588
[57]	eval-auc:0.910794	train-auc:0.966703
[58]	eval-auc:0.911059	train-auc:0.966934
[59]	eval-auc:0.911076	train-auc:0.96708
[60]	eval-auc:0.910823	train-auc:0.967371
[61]	eval-auc:0.911047	train-auc:0.96787
[62]	eval-auc:0.911502	train-auc:0.96834
[63]	eval-auc:0.912839	train-auc:0.968949
[64]	eval-auc:0.913187	train-auc:0.969706
[65]	eval-auc:0.913662	train-auc:0.970076
[66]	eval-auc:0.913552	train-auc:0.970497
[67]	eval-auc:0.913271	train-auc:0.970919
[68]	eval-auc:0.913458	train-auc:0.971182
[69]	eval-auc:0.913458	train-auc:0.971419
[70]	eval-auc:0.913291	train-auc:0.971667
[71]	eval-auc:0.913656	train-auc:0.971884
[72]	eval-auc:0.91411	train-auc:0.972523
[73]	eval-auc:0.914024	train-auc:0.973027
[74]	eval-auc:0.91399	train-auc:0.973126
[75]	eval-auc:0.914059	train-auc:0.973415
[76]	eval-auc:0.914953	train-auc:0.973651
[77]	eval-auc:0.914976	train-auc:0.973998
[78]	eval-auc:0.915123	train-auc:0.974454
[79]	eval-auc:0.914712	train-auc:0.975192
```

```
[80]	eval-auc:0.914729	train-auc:0.975625
[81]	eval-auc:0.914568	train-auc:0.976236
[82]	eval-auc:0.914576	train-auc:0.976249
[83]	eval-auc:0.914576	train-auc:0.976395
[84]	eval-auc:0.914131	train-auc:0.976689
[85]	eval-auc:0.914309	train-auc:0.97674
[86]	eval-auc:0.914407	train-auc:0.977268
[87]	eval-auc:0.91443	train-auc:0.977814
[88]	eval-auc:0.915039	train-auc:0.978037
[89]	eval-auc:0.915034	train-auc:0.978582
[90]	eval-auc:0.915014	train-auc:0.978975
[91]	eval-auc:0.914438	train-auc:0.979451
[92]	eval-auc:0.914548	train-auc:0.979831
[93]	eval-auc:0.915062	train-auc:0.98008
[94]	eval-auc:0.914804	train-auc:0.980339
[95]	eval-auc:0.914864	train-auc:0.980585
[96]	eval-auc:0.914976	train-auc:0.98073
[97]	eval-auc:0.914973	train-auc:0.980925
[98]	eval-auc:0.914674	train-auc:0.981273
[99]	eval-auc:0.914326	train-auc:0.981357
[100]	eval-auc:0.913961	train-auc:0.981714
[101]	eval-auc:0.913952	train-auc:0.982042
[102]	eval-auc:0.913877	train-auc:0.982463
[103]	eval-auc:0.91401	train-auc:0.982884
[104]	eval-auc:0.913564	train-auc:0.983217
[105]	eval-auc:0.913411	train-auc:0.983539
[106]	eval-auc:0.913245	train-auc:0.984032
[107]	eval-auc:0.913161	train-auc:0.984507
[108]	eval-auc:0.912879	train-auc:0.98477
[109]	eval-auc:0.912623	train-auc:0.98514
[110]	eval-auc:0.912796	train-auc:0.985322
[111]	eval-auc:0.913325	train-auc:0.985503
[112]	eval-auc:0.913294	train-auc:0.985796
[113]	eval-auc:0.913601	train-auc:0.986048
[114]	eval-auc:0.914041	train-auc:0.986233
[115]	eval-auc:0.913734	train-auc:0.986426
[116]	eval-auc:0.913472	train-auc:0.986495
[117]	eval-auc:0.913565	train-auc:0.986901
[118]	eval-auc:0.913514	train-auc:0.987268
[119]	eval-auc:0.913404	train-auc:0.987481
[120]	eval-auc:0.913163	train-auc:0.98764
[121]	eval-auc:0.912674	train-auc:0.98779
[122]	eval-auc:0.912818	train-auc:0.987881
[123]	eval-auc:0.912772	train-auc:0.988072
[124]	eval-auc:0.912774	train-auc:0.988223
[125]	eval-auc:0.913432	train-auc:0.98833
[126]	eval-auc:0.913504	train-auc:0.988357
[127]	eval-auc:0.91365	train-auc:0.9887
[128]	eval-auc:0.913285	train-auc:0.988878
[129]	eval-auc:0.912917	train-auc:0.989188
[130]	eval-auc:0.91275	train-auc:0.989312
[131]	eval-auc:0.912434	train-auc:0.989549
[132]	eval-auc:0.912739	train-auc:0.98971
[133]	eval-auc:0.912816	train-auc:0.989802
[134]	eval-auc:0.912552	train-auc:0.989806
[135]	eval-auc:0.91298	train-auc:0.98998
[136]	eval-auc:0.913144	train-auc:0.990115
[137]	eval-auc:0.913213	train-auc:0.990296
[138]	eval-auc:0.913322	train-auc:0.990307
[139]	eval-auc:0.913202	train-auc:0.990428
[140]	eval-auc:0.91342	train-auc:0.990678
[141]	eval-auc:0.913371	train-auc:0.990774
[142]	eval-auc:0.91397	train-auc:0.990966
[143]	eval-auc:0.914171	train-auc:0.991238
```

```
[144]	eval-auc:0.91397	train-auc:0.991414
[145]	eval-auc:0.91401	train-auc:0.99149
[146]	eval-auc:0.9139	train-auc:0.991781
[147]	eval-auc:0.913967	train-auc:0.991849
[148]	eval-auc:0.914228	train-auc:0.99188
[149]	eval-auc:0.914294	train-auc:0.992087
[150]	eval-auc:0.914067	train-auc:0.992253
[151]	eval-auc:0.914565	train-auc:0.992425
[152]	eval-auc:0.914332	train-auc:0.992583
[153]	eval-auc:0.914596	train-auc:0.992739
[154]	eval-auc:0.914645	train-auc:0.992979
[155]	eval-auc:0.91472	train-auc:0.993045
[156]	eval-auc:0.914766	train-auc:0.993114
[157]	eval-auc:0.914389	train-auc:0.993178
[158]	eval-auc:0.914231	train-auc:0.99341
[159]	eval-auc:0.914631	train-auc:0.993534
[160]	eval-auc:0.91453	train-auc:0.993744
[161]	eval-auc:0.913955	train-auc:0.993844
[162]	eval-auc:0.91384	train-auc:0.993891
[163]	eval-auc:0.91334	train-auc:0.993993
[164]	eval-auc:0.913207	train-auc:0.994064
[165]	eval-auc:0.912971	train-auc:0.994087
[166]	eval-auc:0.91279	train-auc:0.994177
[167]	eval-auc:0.912641	train-auc:0.994285
[168]	eval-auc:0.91229	train-auc:0.994369
[169]	eval-auc:0.912413	train-auc:0.994427
[170]	eval-auc:0.91208	train-auc:0.99451
[171]	eval-auc:0.911873	train-auc:0.99461
[172]	eval-auc:0.911758	train-auc:0.994659
[173]	eval-auc:0.912005	train-auc:0.994814
[174]	eval-auc:0.911973	train-auc:0.994932
[175]	eval-auc:0.911706	train-auc:0.995033
[176]	eval-auc:0.911677	train-auc:0.995099
[177]	eval-auc:0.911542	train-auc:0.995217
[178]	eval-auc:0.911257	train-auc:0.995333
[179]	eval-auc:0.911251	train-auc:0.995369
[180]	eval-auc:0.911372	train-auc:0.995495
[181]	eval-auc:0.911424	train-auc:0.995529
[182]	eval-auc:0.911323	train-auc:0.995625
[183]	eval-auc:0.911338	train-auc:0.995676
[184]	eval-auc:0.911162	train-auc:0.995733
[185]	eval-auc:0.911323	train-auc:0.99574
[186]	eval-auc:0.911217	train-auc:0.995766
[187]	eval-auc:0.911349	train-auc:0.995791
[188]	eval-auc:0.911838	train-auc:0.995835
[189]	eval-auc:0.911821	train-auc:0.995996
[190]	eval-auc:0.911392	train-auc:0.996046
[191]	eval-auc:0.911657	train-auc:0.996067
[192]	eval-auc:0.911666	train-auc:0.996119
[193]	eval-auc:0.9117	train-auc:0.996124
[194]	eval-auc:0.911706	train-auc:0.996237
[195]	eval-auc:0.91162	train-auc:0.996395
[196]	eval-auc:0.911519	train-auc:0.996499
[197]	eval-auc:0.911364	train-auc:0.996578
[198]	eval-auc:0.91103	train-auc:0.996764
[199]	eval-auc:0.911493	train-auc:0.996839
[200]	eval-auc:0.911387	train-auc:0.996847
[201]	eval-auc:0.91128	train-auc:0.99688
[202]	eval-auc:0.911116	train-auc:0.996968
[203]	eval-auc:0.911375	train-auc:0.997098
[204]	eval-auc:0.911413	train-auc:0.997213
[205]	eval-auc:0.911404	train-auc:0.997273
[206]	eval-auc:0.911758	train-auc:0.997307
[207]	eval-auc:0.911755	train-auc:0.997325
```

```
[208]	eval-auc:0.911775	train-auc:0.997392
[209]	eval-auc:0.911283	train-auc:0.997469
[210]	eval-auc:0.911323	train-auc:0.997464
[211]	eval-auc:0.91097	train-auc:0.997533
[212]	eval-auc:0.910869	train-auc:0.997584
[213]	eval-auc:0.91086	train-auc:0.997643
[214]	eval-auc:0.910509	train-auc:0.997707
[215]	eval-auc:0.910734	train-auc:0.997765
[216]	eval-auc:0.910662	train-auc:0.997818
[217]	eval-auc:0.910765	train-auc:0.997831
[218]	eval-auc:0.910852	train-auc:0.997916
[219]	eval-auc:0.910929	train-auc:0.997948
[220]	eval-auc:0.910826	train-auc:0.997963
[221]	eval-auc:0.910906	train-auc:0.997982
[222]	eval-auc:0.910714	train-auc:0.997974
[223]	eval-auc:0.910811	train-auc:0.998037
[224]	eval-auc:0.910734	train-auc:0.99805
[225]	eval-auc:0.910734	train-auc:0.998053
[226]	eval-auc:0.910659	train-auc:0.998092
[227]	eval-auc:0.910512	train-auc:0.998103
[228]	eval-auc:0.910386	train-auc:0.99811
[229]	eval-auc:0.910478	train-auc:0.99815
[230]	eval-auc:0.910415	train-auc:0.998232
[231]	eval-auc:0.91032	train-auc:0.998274
[232]	eval-auc:0.910242	train-auc:0.99828
[233]	eval-auc:0.910627	train-auc:0.99834
[234]	eval-auc:0.910553	train-auc:0.998371
[235]	eval-auc:0.910003	train-auc:0.99841
[236]	eval-auc:0.910058	train-auc:0.998438
[237]	eval-auc:0.910256	train-auc:0.998543
[238]	eval-auc:0.910187	train-auc:0.99856
[239]	eval-auc:0.910276	train-auc:0.998609
[240]	eval-auc:0.910285	train-auc:0.998617
[241]	eval-auc:0.910075	train-auc:0.998671
[242]	eval-auc:0.909747	train-auc:0.998732
[243]	eval-auc:0.909773	train-auc:0.99878
[244]	eval-auc:0.909629	train-auc:0.998835
[245]	eval-auc:0.909678	train-auc:0.998866
[246]	eval-auc:0.909822	train-auc:0.998892
[247]	eval-auc:0.909923	train-auc:0.998888
[248]	eval-auc:0.909908	train-auc:0.998873
Stopping. Best iteration:
[246]	eval-auc:0.909822	train-auc:0.998892
```

In [61]:

```
train.head()
```

Out[61]:

| | Id | Age | Marital | Education | Default | Balance | HHInsurance | CarLoan | Communication | La |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 32 | 2 | 3 | 0 | -0.089700 | 1 | 0 | 2 | 28 |
| 1 | 2 | 32 | 1 | 1 | 0 | -0.107359 | 1 | 0 | 1 | 26 |
| 2 | 3 | 29 | 2 | 3 | 0 | -0.255179 | 1 | 0 | 0 | 3 |
| 3 | 4 | 25 | 2 | 1 | 0 | -0.330371 | 1 | 0 | 0 | 11 |
| 4 | 5 | 30 | 1 | 3 | 0 | 0.330692 | 0 | 0 | 0 | 3 |

モデル作成

In [62]:
```
def make_model(train, cat_cols, dtrain, dval):
    eval_list = [(dval, 'eval'), (dtrain, 'train')]
    param = {'max_depth': 4,
             'min_child_weight' : 1,
             'gamma' : 1,
             'subsample' : 1,
             'colsample_bytree' : 1,
             'alpha' : 0.5,
             'labmda' : 0.5,
             'nthread' : 5,
             'eta': 0.15,
             'silent': 1,
             'objective': 'binary:logistic',
             'eval_metric' : 'auc'}

    num_round = 300
    bst = xgb.train(param, dtrain, num_round, eval_list, early_stoppin
g_rounds=2)
    return bst
```

In [63]:
```
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
le = LabelEncoder()
for col in cat_cols:
    train[col] = le.fit_transform(train[col])
    test[col] = le.fit_transform(test[col])
```

In [64]:
```
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
X = train
y = train.CarInsurance
X_train, X_test, y_train, y_test = train_test_split(X, y,
                                                    test_size=0.3, ran
dom_state=0)
```

In [65]:
```
test_id = X_test.Id.values
test_carInsurance = X_test.CarInsurance.values
X_train.drop(['Id', 'CarInsurance'], axis=1, inplace=True)
X_test.drop(['Id', 'CarInsurance'], axis=1, inplace=True)
```

```
/opt/conda/lib/python3.6/site-packages/pandas/core/frame.py:3697: Sett
ingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-
docs/stable/indexing.html#indexing-view-versus-copy
  errors=errors)
```

In [66]:
```
import xgboost as xgb
dtrain = xgb.DMatrix(X_train, y_train)
dval = xgb.DMatrix(X_test, y_test)
```

In [67]:

```
cat_cols = ['Marital', 'Education', 'Communication',
            'LastContactMonth', 'Outcome']

bst = make_model(train, cat_cols, dtrain, dval)
```

```
[0]	eval-auc:0.843711	train-auc:0.873992
Multiple eval metrics have been passed: 'train-auc' will be used for e
arly stopping.

Will train until train-auc hasn't improved in 2 rounds.
[1]	eval-auc:0.857401	train-auc:0.888068
[2]	eval-auc:0.858789	train-auc:0.889743
[3]	eval-auc:0.868222	train-auc:0.898518
[4]	eval-auc:0.86839	train-auc:0.900772
[5]	eval-auc:0.868581	train-auc:0.902292
[6]	eval-auc:0.869914	train-auc:0.90469
[7]	eval-auc:0.87459	train-auc:0.908159
[8]	eval-auc:0.880214	train-auc:0.911055
[9]	eval-auc:0.882404	train-auc:0.914217
[10]	eval-auc:0.882466	train-auc:0.914551
[11]	eval-auc:0.883489	train-auc:0.915835
[12]	eval-auc:0.885043	train-auc:0.918721
[13]	eval-auc:0.885757	train-auc:0.919617
[14]	eval-auc:0.887242	train-auc:0.920942
[15]	eval-auc:0.888807	train-auc:0.922525
[16]	eval-auc:0.890478	train-auc:0.924871
[17]	eval-auc:0.891534	train-auc:0.926951
[18]	eval-auc:0.892957	train-auc:0.928164
[19]	eval-auc:0.895962	train-auc:0.930997
[20]	eval-auc:0.897326	train-auc:0.932759
[21]	eval-auc:0.897847	train-auc:0.93358
[22]	eval-auc:0.898956	train-auc:0.934713
[23]	eval-auc:0.901668	train-auc:0.937263
[24]	eval-auc:0.901987	train-auc:0.938511
[25]	eval-auc:0.902735	train-auc:0.939382
[26]	eval-auc:0.903814	train-auc:0.940139
[27]	eval-auc:0.904308	train-auc:0.941018
[28]	eval-auc:0.904941	train-auc:0.943245
[29]	eval-auc:0.90524	train-auc:0.943767
[30]	eval-auc:0.906739	train-auc:0.9445
[31]	eval-auc:0.90758	train-auc:0.945211
[32]	eval-auc:0.907701	train-auc:0.945981
[33]	eval-auc:0.907763	train-auc:0.946823
[34]	eval-auc:0.908723	train-auc:0.948778
[35]	eval-auc:0.909281	train-auc:0.949077
[36]	eval-auc:0.909593	train-auc:0.949645
[37]	eval-auc:0.909997	train-auc:0.950733
[38]	eval-auc:0.909675	train-auc:0.951515
[39]	eval-auc:0.910371	train-auc:0.952047
[40]	eval-auc:0.910199	train-auc:0.953082
[41]	eval-auc:0.909999	train-auc:0.953735
[42]	eval-auc:0.910571	train-auc:0.954585
[43]	eval-auc:0.910761	train-auc:0.954991
[44]	eval-auc:0.910699	train-auc:0.95594
[45]	eval-auc:0.910288	train-auc:0.956393
[46]	eval-auc:0.910279	train-auc:0.956551
[47]	eval-auc:0.910285	train-auc:0.956867
[48]	eval-auc:0.910929	train-auc:0.957337
[49]	eval-auc:0.911182	train-auc:0.958014
[50]	eval-auc:0.911919	train-auc:0.958921
[51]	eval-auc:0.911936	train-auc:0.9594
[52]	eval-auc:0.912264	train-auc:0.960366
[53]	eval-auc:0.912276	train-auc:0.960858
```

```
[53]	eval-auc:0.912376	train-auc:0.960858
[54]	eval-auc:0.913394	train-auc:0.961732
[55]	eval-auc:0.913947	train-auc:0.962622
[56]	eval-auc:0.914018	train-auc:0.963339
[57]	eval-auc:0.914254	train-auc:0.963866
[58]	eval-auc:0.914999	train-auc:0.964656
[59]	eval-auc:0.914545	train-auc:0.965208
[60]	eval-auc:0.914818	train-auc:0.966004
[61]	eval-auc:0.91485	train-auc:0.966996
[62]	eval-auc:0.915008	train-auc:0.967428
[63]	eval-auc:0.915643	train-auc:0.967847
[64]	eval-auc:0.915402	train-auc:0.968195
[65]	eval-auc:0.915764	train-auc:0.968901
[66]	eval-auc:0.915917	train-auc:0.969333
[67]	eval-auc:0.916017	train-auc:0.970077
[68]	eval-auc:0.916072	train-auc:0.970568
[69]	eval-auc:0.916069	train-auc:0.971027
[70]	eval-auc:0.915563	train-auc:0.971239
[71]	eval-auc:0.915802	train-auc:0.971581
[72]	eval-auc:0.915678	train-auc:0.971815
[73]	eval-auc:0.915707	train-auc:0.972114
[74]	eval-auc:0.916662	train-auc:0.972333
[75]	eval-auc:0.91776	train-auc:0.972977
[76]	eval-auc:0.918137	train-auc:0.973599
[77]	eval-auc:0.918894	train-auc:0.974045
[78]	eval-auc:0.918994	train-auc:0.974389
[79]	eval-auc:0.918779	train-auc:0.974714
[80]	eval-auc:0.918813	train-auc:0.974739
[81]	eval-auc:0.919342	train-auc:0.975209
[82]	eval-auc:0.919368	train-auc:0.975879
[83]	eval-auc:0.918925	train-auc:0.976141
[84]	eval-auc:0.918868	train-auc:0.976303
[85]	eval-auc:0.918738	train-auc:0.976604
[86]	eval-auc:0.918839	train-auc:0.976744
[87]	eval-auc:0.918764	train-auc:0.97708
[88]	eval-auc:0.918865	train-auc:0.977625
[89]	eval-auc:0.919029	train-auc:0.97817
[90]	eval-auc:0.918954	train-auc:0.978305
[91]	eval-auc:0.918856	train-auc:0.978581
[92]	eval-auc:0.918894	train-auc:0.978676
[93]	eval-auc:0.918684	train-auc:0.978837
[94]	eval-auc:0.918525	train-auc:0.978995
[95]	eval-auc:0.918468	train-auc:0.979489
[96]	eval-auc:0.918707	train-auc:0.979805
[97]	eval-auc:0.919011	train-auc:0.980142
[98]	eval-auc:0.919109	train-auc:0.980201
[99]	eval-auc:0.919104	train-auc:0.980199
[100]	eval-auc:0.919457	train-auc:0.980567
[101]	eval-auc:0.919472	train-auc:0.981132
[102]	eval-auc:0.919627	train-auc:0.981391
[103]	eval-auc:0.919673	train-auc:0.981657
[104]	eval-auc:0.919748	train-auc:0.981915
[105]	eval-auc:0.91965	train-auc:0.982071
[106]	eval-auc:0.920076	train-auc:0.982203
[107]	eval-auc:0.920622	train-auc:0.982273
[108]	eval-auc:0.920507	train-auc:0.982462
[109]	eval-auc:0.920455	train-auc:0.982735
[110]	eval-auc:0.920455	train-auc:0.982735
[111]	eval-auc:0.920455	train-auc:0.982735
Stopping. Best iteration:
[109]	eval-auc:0.920455	train-auc:0.982735
```

In [68]:

In [68]:

```
dtest = xgb.DMatrix(X_test)
predict = bst.predict(dtest)
```

In [69]:

```
submissions = pd.DataFrame({"Id" : test_id, "PredValue" : predict, 'Ca
rInsurance' : test_carInsurance})
```

In [70]:

```
submissions.to_csv("./submissions.csv", index=False)
```

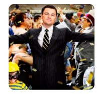
In [71]:

```
def make_model_optuna(trial):
    from sklearn.preprocessing import LabelEncoder
    from sklearn.model_selection import train_test_split
    import xgboost as xgb
    import sklearn.datasets
    import sklearn.metrics

    le = LabelEncoder()
    for col in cat_cols:
        train[col] = le.fit_transform(train[col])
        test[col] = le.fit_transform(test[col])

    X = train.drop(['Id', 'CarInsurance'], axis=1)
    y = train.CarInsurance
    X_train, X_test, y_train, y_test = train_test_split(X, y,
                                                        test_size=0.3,
random_state=0)

    dtrain = xgb.DMatrix(X_train, y_train)
    dval = xgb.DMatrix(X_test, y_test)
    # eval_list = [(dval, 'eval'), (dtrain, 'train')]

    n_round = trial.suggest_int('n_round', 1, 9)
    param = {'silent': 1,
             'objective': 'binary:logistic',
             'booster': trial.suggest_categorical('booster', ['gbtree'
, 'gblinear', 'dart']),
             'lambda': trial.suggest_loguniform('lambda', 1e-8, 1.0),
             'alpha': trial.suggest_loguniform('alpha', 1e-8, 1.0),
             'eval_metric' : 'auc'
             }

    if param['booster'] == 'gbtree' or param['booster'] == 'dart':
        param['max_depth'] = trial.suggest_int('max_depth', 1, 9)
        param['ets'] = trial.suggest_loguniform('eta', 1e-8, 1.0)
        param['gamma'] = trial.suggest_loguniform('gamma', 1e-8, 1.0)
        param['grow_policy'] = trial.suggest_categorical('grow_policy'
, ['depthwise', 'lossguide'])

    if param['booster'] == 'dart':
        param['sample_type'] = trial.suggest_categorical('sample_type'
, ['uniform', 'weighted'])
```

Data

**Data Sources**

- Car Insurance Cold Calls
  - carInsurance_test.csv 19 columns
  - carInsurance_train.csv 19 columns
  - DSS_DMC_Description.pdf



## Car Insurance Cold Calls

**We help the guys and girls at the front to get out of Cold Call Hell**

Last Updated: 2 years ago (Version 1)

**About this Dataset**

### Introduction

Here you find a very simple, beginner-friendly data set. No sparse matrices, no fancy tools needed to understand what's going on. Just a couple of rows and columns. Super simple stuff. As explained below, this data set is used for a competition. As it turns out, this competition tends to reveal a common truth in data science: KISS - Keep It Simple Stupid

What is so special about this data set is, given it's simplicity, it pays off to use "simple" classifiers as well. This year's competition was won by a C5.0 . Can you do better?

### Description

We are looking at cold call results. Turns out, same salespeople called existing insurance customers up and tried to sell car insurance. What you have are details about the called customers. Their age, job, marital status, whether the have home insurance, a car loan, etc. As I said, super simple.

What I would love to see is some of you applying some crazy XGBoost classifiers, which we can square off against some logistic regressions. It would be curious to see what comes out on top. Thank you for your time, I hope you enjoy using the data set.

### Acknowledgements

Thanks goes to the Decision Science and Systems Chair of Technical University of Munich (TUM) for getting the data set

Output Files

New Notebook | Download All

**Output Files**

- submissions.csv

**About this file**

This file was created from a Kernel, it does not have a description.

submissions.csv

| 1 | Id | PredValue | CarInsurance |
|---|---|---|---|
| 2 | 2231 | 0.014460921 | 0 |
| 3 | 669 | 0.421857 | 1 |
| 4 | 3617 | 0.059944265 | 0 |
| 5 | 2364 | 0.28126207 | 0 |
| 6 | 143 | 0.008655553 | 0 |

| | | | |
|---|---|---|---|
| 7 | 539 | 0.61982554 | 1 |
| 8 | 1792 | 0.8545701 | 0 |
| 9 | 411 | 0.57173336 | 0 |
| 10 | 1151 | 0.027505957 | 0 |
| 11 | 1033 | 0.9008635 | 1 |
| 12 | 2179 | 0.01949213 | 0 |
| 13 | 225 | 0.0051516746 | 0 |
| 14 | 2801 | 0.0054382123 | 0 |
| 15 | 2282 | 0.24459662 | 0 |
| 16 | 3311 | 0.22591984 | 0 |
| 17 | 1747 | 0.35132027 | 1 |
| 18 | 2859 | 0.6926998 | 1 |
| 19 | 2407 | 0.79104865 | 0 |
| 20 | 3660 | 0.74910945 | 1 |
| 21 | 3028 | 0.005728394 | 0 |
| 22 | 3464 | 0.9128454 | 1 |
| 23 | 1372 | 0.83979803 | 1 |
| 24 | 966 | 0.7271425 | 1 |
| 25 | 1016 | 0.005733377 | 0 |
| 26 | 911 | 0.00232833 | 0 |

Comments (0)

Please sign in to leave a comment.