

1.) Import the Credit Card Fraud Data From CCLE

```
import pandas as pd
from google.colab import drive
import matplotlib.pyplot as plt
import numpy as np
```

```
drive.mount('/content/gdrive/', force_remount = True)
```

```
Mounted at /content/gdrive/
```

```
df = pd.read_csv("/content/gdrive/MyDrive/ECON441B/fraudTest.csv")
```

```
df.head()
```

	Unnamed: 0	trans_date_trans_time	cc_num	merchant	category	amt	first	last	gender	street	...	lat	lon
0	0	2020-06-21 12:14:25	2291163933867244	fraud_Kirlin and Sons	personal_care	2.86	Jeff	Elliott	M	351 Darlene Green	...	33.9659	-80.93
1	1	2020-06-21 12:14:33	3573030041201292	fraud_Sporer-Keebler	personal_care	29.84	Joanne	Williams	F	3638 Marsh Union	...	40.3207	-110.43
2	2	2020-06-21 12:14:53	3598215285024754	fraud_Swaniawski, Nitzsche and Welch	health_fitness	41.28	Ashley	Lopez	F	9333 Valentine Point	...	40.6729	-73.53
3	3	2020-06-21 12:15:15	3591919803438423	fraud_Haley Group	misc_pos	60.05	Brian	Williams	M	32941 Krystal Mill Apt. 552	...	28.5697	-80.8
4	4	2020-06-21 12:15:17	3526826139003047	fraud_Johnston-Casper	travel	3.19	Nathan	Massey	M	5783 Evan Roads Apt. 465	...	44.2529	-85.0

5 rows × 23 columns

2.) Select four columns to use as features (one just be trans_date_trans)

```
df_select = df[["trans_date_trans_time", "category", "amt", "city_pop", "is_fraud"]]
```

```
df_select.columns
```

```
Index(['trans_date_trans_time', 'category', 'amt', 'city_pop', 'is_fraud'], dtype='object')
```

3.) Create a your own variable out of trans_date. Create dummies for factor vars

```
type(df_select["trans_date_trans_time"][0])
```

```
str
```

```
df_select["trans_date_trans_time"] = pd.to_datetime(df_select["trans_date_trans_time"])
```

```
<ipython-input-9-99f721e4ce0f>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead
```

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy

```
df_select["trans_date_trans_time"] = pd.to_datetime(df_select["trans_date_trans_time"])
```

```
dir(df_select["trans_date_trans_time"][0])
```

```
'freq',
'freqstr',
'fromisocalendar',
'fromisoformat',
'fromordinal',
'fromtimestamp',
'hour',
'is_leap_year',
'is_month_end',
'is_month_start',
'is_quarter_end',
'is_quarter_start',
'is_year_end',
'is_year_start',
'isocalendar',
'isoformat',
'isoweekday',
'max',
'microsecond',
'min',
'minute',
'month',
'month_name',
'nanosecond',
'normalize',
'now',
'quarter',
'replace',
'resolution',
'round',
'second',
'strftime',
'strptime',
'time',
'timestamp',
'timetuple',
'timetz',
'to_datetime64',
'to_julian_date',
'to_numpy',
'to_period',
'to_pydatetime',
'today',
'toordinal',
'tz',
'tz_convert',
'tz_localize',
'tzinfo',
'tzname',
'utcfromtimestamp',
'utcnow',
'utcoffset',
'utctimetuple',
'value',
'week',
'weekday',
'weekofyear',
'year']
```

```
X = pd.get_dummies(df_select, ["category"]).drop(["trans_date_trans_time", "is_fraud"], axis = 1)
y = df["is_fraud"]
```

```
X.head()
```

	amt	city_pop	category_entertainment	category_food_dining	category_gas_transport	category_grocery_net	category_grocery_pos	category
0	2.86	333497	0	0	0	0	0	0
1	29.84	302	0	0	0	0	0	0
2	41.28	34496	0	0	0	0	0	0
3	60.05	54767	0	0	0	0	0	0
4	3.19	1126	0	0	0	0	0	0

XXX SKIP THIS WE WILL TALK ABOUT NEXT CLASS

```
resample_X = X
resample_y = y
```

5.) Train a Logistic regression.

```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
X_normalized = scaler.fit_transform(resample_X)
```

```
from sklearn.linear_model import LogisticRegression
```

```
log_reg = LogisticRegression().fit(X_normalized, resample_y)
```

6.) The company you are working for wants to target at a False Positive rate of 5% what threshold should you use? (Use oversampled data)

```
from sklearn.metrics import confusion_matrix
```

```
clf = LogisticRegression()
clf.fit(X_normalized, resample_y)
```

```
LogisticRegression()
```

```
y_prob = clf.predict_proba(X_normalized)
target_fn_percentage = 5
threshold = np.percentile(y_prob[:,1],100-target_fn_percentage)
y_pred = (y_prob[:,1]>threshold).astype(int)
confusion_matrix(y, y_pred)
```

```
array([[527247, 26327],
       [ 686, 1459]])
```

7.) If the company makes .02*amt on True transactions and loses -amt on False (Use original data)

```
df_temp = df_select.copy()
```

```
df_temp["pred"] = log_reg.predict(resample_X)
```

```
/usr/local/lib/python3.8/dist-packages/sklearn/base.py:443: UserWarning: X has feature names, but LogisticRegression was fitted without feature names
  warnings.warn(
```

```
df_temp = df_temp[["pred", "is_fraud", "amt"]]
```

```
df_temp.head()
```

	pred	is_fraud	amt
0	0	0	2.86
1	0	0	29.84
2	0	0	41.28
3	0	0	60.05
4	0	0	3.19

```
dataframe = df_temp.loc[df_temp['pred']==0]
dataframe.dropna(axis = 0, how='any',inplace = True)
dataframe = dataframe.reset_index()
```

```
/usr/local/lib/python3.8/dist-packages/pandas/util/_decorators.py:311: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame
```

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy

```
return func(*args, **kwargs)
```

```
p=0
for i in range(0,len(dataframe)):
    if dataframe.loc[i,'is_fraud']==0:
        p= p-dataframe.loc[i,'amt']
print("Predicted profit is:",p)

Predicted profit is: -31980839.799999993
```

8.) Using Logistic Regression Lasso to inform you. Would you use the selected features in a trusted prediction model?

```
LogisticRegression("l1")

LogisticRegression(penalty='l1')

from sklearn.linear_model import LogisticRegressionCV
clf1 = LogisticRegressionCV(cv=5,penalty='l1',solver = 'liblinear')
clf1.fit(X,y)
yhat = clf1.predict(X)
clf1.coef_

array([[ 3.67459432e-04, -7.30539401e-06,  0.00000000e+00,
         0.00000000e+00,  0.00000000e+00,  0.00000000e+00,
         0.00000000e+00,  0.00000000e+00,  0.00000000e+00,
         0.00000000e+00,  0.00000000e+00,  0.00000000e+00,
         0.00000000e+00,  0.00000000e+00,  0.00000000e+00,
         0.00000000e+00])
```

Since most of the variables go to zero, so the selected features are not in a trusted prediction model.