In [19]:
```
import pandas as pd

dtypes = {'state': str, 'approved_risk_class': str, 'alcohol_current_use': str, 'alcoho
          'occupation_2019': str,
          'risky_activities': str, 'risky_behavior_2019': str, 'valid_drivers_license_a
          'hiv': str, 'hiv_ca': str, 'hiv_fl': str,
          'covid': str, 'previous_declined': str, 'previous_decline_2019': str, 'chest_
          'family_history': str,
          'medical_collection_ca_2019': str, 'medical_collection_none': str, 'seizure_d
          'stroke_tia_last_2019': str,
          'disability_audio_visual': str, 'expected_travel_90_days': str, 'expected_tra
          'mental_health_missed_work': str, 'seizure_car_accident': str, 'tobacco_2019'
          'chest_pain_angina': str, 'chest_pain_diagnosed': str, 'med_conditions': str,
         'stroke_count': float, 'stroke_last': str, 'stroke_diagnosis': str, 'stroke_di
          'dui_count': float, 'expected_travel': str,
         'expected_travel_fl': str, 'mental_health_hospitalized': str, 'scuba_130ft': s
         }
```

In [20]:
```
df = pd.read_csv(r'C:\Users\milan\Documents\GitHub\DSC-680\approved_risk_class.csv', dt
```

In [21]:
```
df.shape
```

Out[21]:
```
(173557, 151)
```

In [22]:
```
df.head()
```

Out[22]:

	quote_id	application_id	date	age	anb	gender	height	weight	state	approved_risk_cl
0	52df592f-9c61-40ab-8dc3-5663562b6346	81ce35d2-f376-4384-9ead-eb7359c393fa	2021-04-08	48.534878	49	male	68.0	195.0	TX	N
1	8722c617-eea8-4008-a735-1970dd1bd9cb	4acdc655-6fce-4c13-9c2e-1bac29df778e	2021-01-28	42.716825	43	male	67.0	200.0	NM	N
2	39bf00db-16d5-49f2-8007-6eb163c4d1ab	1c12bb4d-8070-4c50-b193-907e4ec0183b	2021-02-02	54.399474	54	male	66.0	193.0	KS	N
3	368be5aa-8bf6-4063-bd1d-4122fafd2093	c2f5fec0-e747-4b74-89fe-fc79ff6bc725	2021-01-18	50.336420	50	female	63.0	140.0	OK	N
4	43a8e77e-6e91-497f-983a-2d09a3c8e870	373879b7-a94f-439f-b4e2-cb068885c06d	2021-03-07	42.725039	43	female	61.0	132.0	IN	N

5 rows × 151 columns

In [23]:
```
# drop fields with no non null values

df.drop(['replacement_ins_company_immutable',
'replacement_ins_policy_number_immutable',
'alcohol_drink_count',
'alcohol_drug_abuse',
'alcohol_monthly_use',
'occupation',
'hiv_multiselect',
'chest_pain_last',
'chest_pain_nitro',
'diabetes_before_forty',
'diabetes_kidney_disease',
'prescribed_insulin',
'medical_collection_none',
'stroke',
'disability_condition',
'disability_condition_other',
'disability_payments',
'disability_payments_2019',
'dui_history',
'government_id',
'depression_diagnosed',
'depression_meds',
'depression_mental_stress',
'prescribed_depression',
'prescribed_quantity_depression',
'claim_hospital_depression',
'work_missed_depression',
'when_depression_diagnosed',
'rx_increase_condition',
'rx_increase_other',
'grand_mal_seizure',
'how_many_seizures',
'prescribed_for_seizures',
'weight_loss_surgery',
'active_military_specify',
'age_diabetes_diagnosed',
'cancer',
'chronic_kidney_disease',
'citizen_base',
'heart_disease',
'legal_alcohol_drugs',
'legal_condition',
'legal_license',
'legal_none',
'liver',
'organ_transplant',
'peripheral_arterial_disease',
'previous_insurance',
'previous_insurance_reason',
'reckless_driving_count',
'skydive_type',
'suicide_depression',
'travel_duration'], axis=1, inplace=True)
```

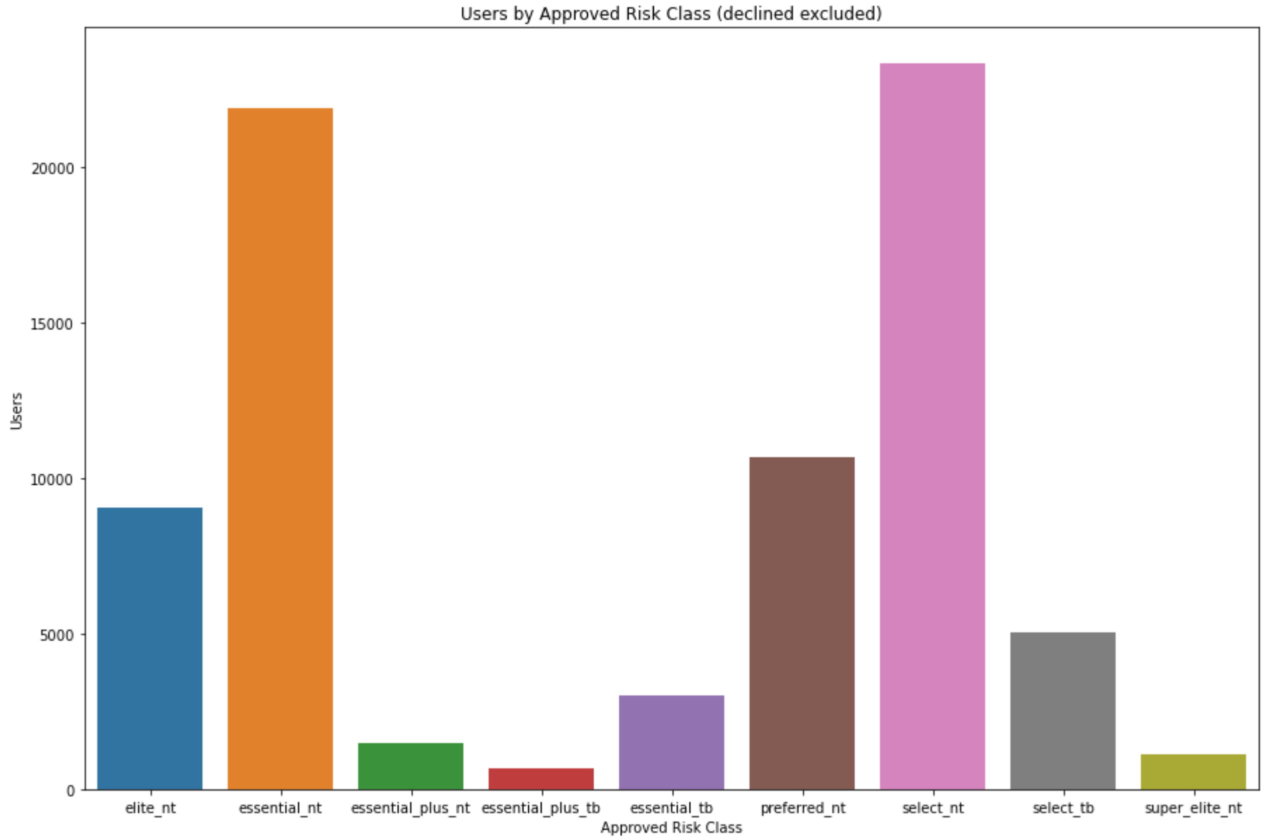
In [24]:
```
df.shape
```

Out[24]:
```
(173557, 98)
```

In [25]:
```
df['approved_risk_class'] = df['approved_risk_class'].fillna('declined')
```

In [33]:
```
import pandas as pd
import matplotlib.pyplot as plt


# Group the data by category and calculate the sum of values
grouped = df.groupby('approved_risk_class')['application_id'].count().reset_index()
print(grouped)

# Create a bar chart from the grouped data
import seaborn as sns
import matplotlib.pyplot as plt

# Create a figure object with the desired size
fig, ax = plt.subplots(figsize=(15,10))

# Create the bar plot
sns.barplot(x="approved_risk_class", y="application_id", data=grouped[grouped['approved

# Customize the plot
ax.set_title("Users by Approved Risk Class (declined excluded)")
ax.set_xlabel("Approved Risk Class")
ax.set_ylabel("Users")

# Display the plot
plt.show()
```

```
  approved_risk_class  application_id
0            declined           97311
1            elite_nt            9042
2        essential_nt           21910
3   essential_plus_nt            1468
4   essential_plus_tb             679
5        essential_tb            2997
6        preferred_nt           10664
7           select_nt           23326
8           select_tb            5039
9      super_elite_nt            1121
```



In [9]:

```
# I am going to remove the super elite and essential plus risk class because there are
# these risk classes are not necessary

# I am also going to remove the tobacco risk classes as they generally follow different
# in terms of risk classification.  A separate model could be created for them.

nt = df[(df['approved_risk_class'] != 'essential_plus_tb') & (df['approved_risk_class']
        (df['approved_risk_class'] != 'select_tb') & (df['approved_risk_class'] != 'sup

nt.shape
```

Out[9]: (163721, 94)

In [10]:

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
~\AppData\Local\Temp/ipykernel_21008/1915129063.py in <module>
      4 
      5 # Group the data by category and calculate the sum of values
----> 6 nt.groupby('approved_risk_class')['application_id'].count()
      7 
      8 # Create a bar chart from the grouped data

~\anaconda3\lib\site-packages\pandas\core\groupby\generic.py in __getitem__(self, key)
   1536                 stacklevel=2,
   1537             )
-> 1538         return super().__getitem__(key)
   1539 
   1540     def _gotitem(self, key, ndim: int, subset=None):
```

```
~\anaconda3\lib\site-packages\pandas\core\base.py in __getitem__(self, key)
    230         else:
    231             if key not in self.obj:
--> 232                 raise KeyError(f"Column not found: {key}")
    233             subset = self.obj[key]
    234             ndim = subset.ndim

KeyError: 'Column not found: application_id'
```

In [ ]:

```
# drop id fields, date field, and anb (same as age)
df.drop(['quote_id', 'application_id', 'date', 'anb'], axis=1, inplace = True)
```

examine outliers: age, weight, height, bmi, income

In [151…

```
df_num = nt.select_dtypes(include = "number")

df_num.isnull().sum()
```

Out[151…

```
age                           0
height                        0
weight                        0
mortality_rate            97311
bmi_app_state                 0
income_app_state              0
household_income         132727
current_ins_value        138843
diabetes_age             153953
alcohol_weekly             9032
alcohol_drink_count_2019 163703
alcohol_monthly_binge    161189
alcohol_monthly_use_2019 163710
marijuana_monthly_count  144581
diabetes_hemoglobin      159226
stroke_count             163418
dui_count                159520
weight_loss_amount       143623
skydive_count            162636
dtype: int64
```

In [152…

```
from sklearn.ensemble import IsolationForest
import numpy as np

num_df = df_num[['age', 'height', 'weight', 'bmi_app_state', 'income_app_state']]

X = num_df.values
# Create an instance of the IsolationForest algorithm
clf = IsolationForest(n_estimators=100, contamination=0.01, random_state=42)

# Fit the model to the data
clf.fit(X)
y_pred = clf.predict(X)

# Print the indices of the predicted outliers
outlier_indices = np.where(y_pred == -1)[0]
outliers_df = num_df.iloc[outlier_indices]
```

```
print(outliers_df)
```

```
             age  height  weight  bmi_app_state  income_app_state
32     54.155801    71.0   215.0      29.983138           3380000
150    55.344052    72.0   173.0      23.460455          11783148
238    43.316427    65.0   185.0      30.782249           2704000
533    54.774568    63.0   185.0      32.767700           2496000
557    38.834473    73.0   165.0      21.766748           7800000
...          ...     ...     ...            ...               ...
172957 50.142029    62.0   140.0      25.603538           2600000
172972 23.020322    64.0   155.0      26.602783           2730000
173016 19.669124    72.0   280.0      37.970679             45614
173024 33.005469    63.0   500.0      88.561350             18720
173244 51.363135    62.0   140.0      25.603538           2908800

[1638 rows x 5 columns]
```

In [153…

```
nt.iloc[outlier_indices]
```

Out[153…

	age	gender	height	weight	state	approved_risk_class	mortality_rate	bmi_app_state	in
32	54.155801	male	71.0	215.0	OK	select_nt	123.0	29.983138	
150	55.344052	male	72.0	173.0	NJ	declined	NaN	23.460455	
238	43.316427	female	65.0	185.0	PA	declined	NaN	30.782249	
533	54.774568	female	63.0	185.0	IL	declined	NaN	32.767700	
557	38.834473	male	73.0	165.0	CA	select_nt	123.0	21.766748	
...	...	...	...	...	...	...	...	...	
172957	50.142029	female	62.0	140.0	FL	declined	NaN	25.603538	
172972	23.020322	male	64.0	155.0	TX	declined	NaN	26.602783	
173016	19.669124	male	72.0	280.0	UT	essential_nt	145.0	37.970679	
173024	33.005469	female	63.0	500.0	VA	declined	NaN	88.561350	
173244	51.363135	female	62.0	140.0	TN	declined	NaN	25.603538	

1638 rows × 94 columns

In [154…

```
# Look at outliers by risk class

outliers = nt.iloc[outlier_indices]
outliers.groupby('approved_risk_class').count().reset_index()
```

Out[154…

	approved_risk_class	age	gender	height	weight	state	mortality_rate	bmi_app_state	income_app_st
0	declined	984	984	984	984	984	0	984	
1	elite_nt	85	85	85	85	85	85	85	
2	essential_nt	229	229	229	229	229	229	229	

	approved_risk_class	age	gender	height	weight	state	mortality_rate	bmi_app_state	income_app_st
3	essential_plus_nt	8	8	8	8	8	8	8	
4	preferred_nt	124	124	124	124	124	124	124	
5	select_nt	208	208	208	208	208	208	208	

6 rows × 94 columns

In [155…

```
# 1% of our data is outliers
outliers.shape[0]/nt.shape[0]
```

Out[155…

```
0.01000482528203468
```

In [156…

```
# drop outliers

nt_clean = nt.drop(nt.index[outlier_indices])
nt_clean.shape
```

Out[156…

```
(162083, 94)
```

check replace insurance and replacement insurance policy number

In [157…

```
ins = nt_clean[nt_clean['replacement_ins'].notnull()]
ins[['replacement_ins', 'replacement_ins_company', 'replacement_ins_policy_number']]
```

Out[157…

	replacement_ins	replacement_ins_company	replacement_ins_policy_number
10	no	NaN	NaN
19	no	NaN	NaN
20	no	NaN	NaN
39	no	NaN	NaN
46	no	NaN	NaN
...	...	...	...
173534	no	NaN	NaN
173536	no	NaN	NaN
173545	no	NaN	NaN
173547	yes	Thrivent	42620
173549	no	NaN	NaN

24515 rows × 3 columns

In [158…

```
# are there any cases where insurance company is not null, but ins is?  No
nt_clean[(nt_clean['replacement_ins'].isna()) & (nt_clean['replacement_ins_company'].no
```

Out[158…

age	gender	height	weight	state	approved_risk_class	mortality_rate	bmi_app_state	income_app_sta

0 rows × 94 columns

In [159…

```
# are there any cases where insurance policy is not null, but ins is?  No
nt_clean[(nt_clean['replacement_ins'].isna()) & (nt_clean['replacement_ins_policy_numbe
```

Out[159…

age	gender	height	weight	state	approved_risk_class	mortality_rate	bmi_app_state	income_app_sta

0 rows × 94 columns

In [160…

```
# drop replacement_ins_company and replacement_ins_policy_number
nt_clean.drop(['replacement_ins_company', 'replacement_ins_policy_number'], axis=1, inp
```

In [161…

```
alcohol_cols = nt_clean.filter(like='alcohol')

alcohol_cols
```

Out[161…

	alcohol_weekly	alcohol_current_use	alcohol_drink_count_2019	alcohol_monthly_binge	alcohol_m
0	0.0	NaN	NaN	NaN	
1	0.0	NaN	NaN	NaN	
2	0.0	NaN	NaN	NaN	
3	NaN	NaN	NaN	NaN	
4	0.0	NaN	NaN	NaN	
...	...	...	...	...	
173552	0.0	NaN	NaN	NaN	
173553	1.0	NaN	NaN	NaN	
173554	0.0	NaN	NaN	NaN	
173555	0.0	NaN	NaN	NaN	
173556	0.0	NaN	NaN	NaN	

162083 rows × 5 columns

In [162…

```
# remove highly correlated and high null columns

nt_clean.drop(['alcohol_current_use', 'alcohol_drink_count_2019', 'alcohol_monthly_bing
                          'alcohol_monthly_use_2019', 'occupation_2019', 'risky_behavio
                         'previous_decline_2019', 'chest_pain_angina', 'chest_pain_diag
```

```
                    'medical_collection_2019', 'stroke_tia_last_2019', 'tobacco_20
                     'criminal_dui_years'], axis=1, inplace=True)
```

In [163…

```
nt_clean.shape
```

Out[163…

```
(162083, 76)
```

In [164…

```
nt_clean = nt_clean[(nt_clean['dui_count'] <= 25) | (nt_clean['dui_count'].isna())]
```

In [165…

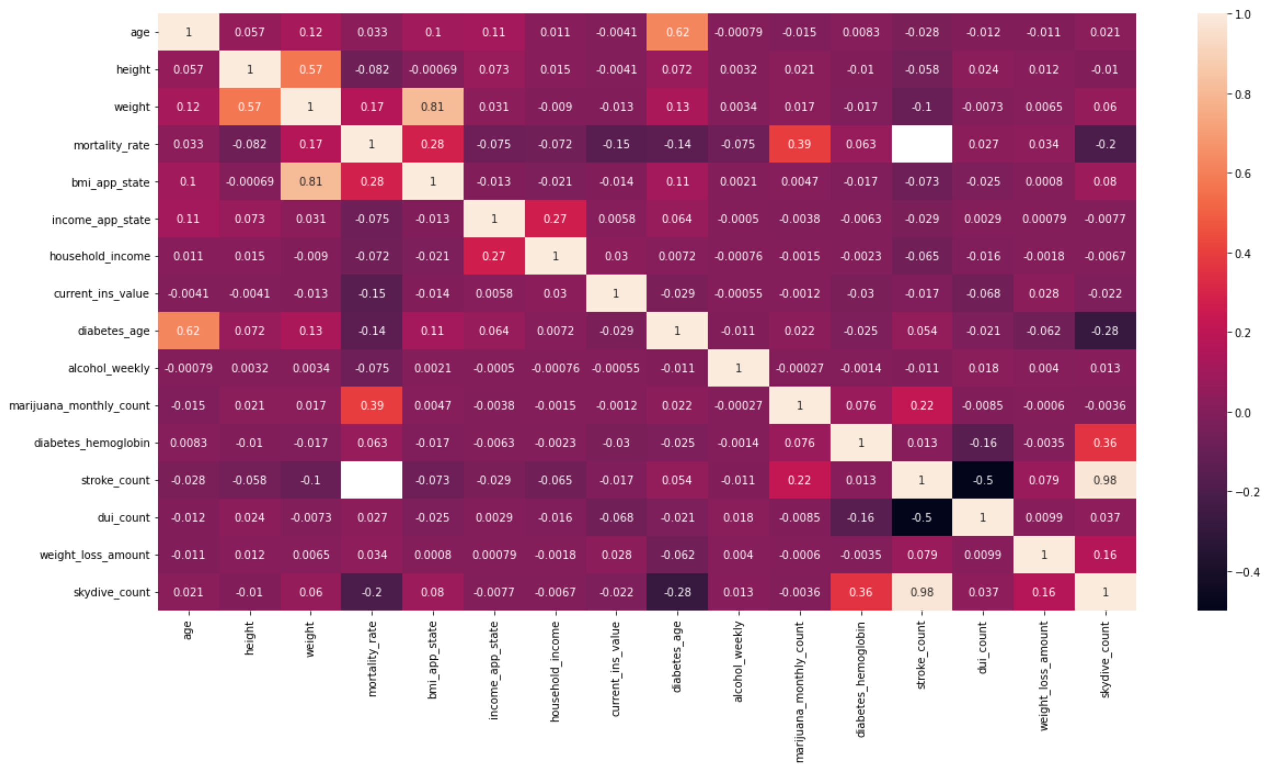
```
nt_clean.shape
```

Out[165…

```
(162060, 76)
```

In [166…

```
df_num = nt_clean.select_dtypes(include='number')
corr_mat = df_num.corr()

import seaborn as sn
import matplotlib.pyplot as plt
plt.rcParams['figure.figsize'] = (20, 10)
sn.heatmap(corr_mat, annot=True)
plt.show()
```



In [167…

```
# ~350 users ineligible by age
nt_clean[(nt_clean['age'] < 18 )| (nt_clean['age'] > 60 ) ]
```

Out[167…

	age	gender	height	weight	state	approved_risk_class	mortality_rate	bmi_app_state	in
129	60.466676	male	71.0	195.0	VA	declined	NaN	27.194009	

	age	gender	height	weight	state	approved_risk_class	mortality_rate	bmi_app_state	in
299	60.384539	female	68.0	125.0	IL	declined	NaN	19.004109	
372	60.365374	male	68.0	140.0	NJ	declined	NaN	21.284602	
2100	60.258595	female	65.0	130.0	TX	declined	NaN	21.630769	
2296	60.121700	male	69.0	153.0	SC	declined	NaN	22.591682	
...	...	...	...	...	...	...	...	...	
171648	60.285974	male	71.0	195.0	CO	declined	NaN	27.194009	
172023	60.373587	male	66.0	180.0	TX	declined	NaN	29.049587	
172923	60.403704	male	69.0	175.0	MI	preferred_nt	90.0	25.840160	
172997	60.348946	male	71.0	205.0	WI	declined	NaN	28.588574	
173224	60.250382	female	66.0	189.0	GA	declined	NaN	30.502066	

352 rows × 76 columns

In [168…

```
#  ~ 900 users ineligible by BMI
nt_clean[(nt_clean['bmi_app_state'] < 18.5 )| (nt_clean['bmi_app_state'] > 40 ) ]
```

Out[168…

	age	gender	height	weight	state	approved_risk_class	mortality_rate	bmi_app_state	in
829	52.118798	female	69.0	300.0	TX	declined	NaN	44.297417	
1648	53.797135	female	66.0	250.0	WA	declined	NaN	40.346648	
4072	49.046866	male	72.0	350.0	IN	declined	NaN	47.463349	
5813	35.576364	male	69.0	330.0	TX	declined	NaN	48.727158	
6470	37.325886	female	66.0	250.0	MA	declined	NaN	40.346648	
...	...	...	...	...	...	...	...	...	
172389	31.822693	female	65.0	245.0	PA	declined	NaN	40.765680	
172969	35.529819	male	71.0	334.0	DE	declined	NaN	46.578457	
173037	43.258931	female	65.0	245.0	PA	declined	NaN	40.765680	
173261	41.933784	female	63.0	270.0	SC	declined	NaN	47.823129	
173327	42.807176	female	62.0	235.0	MI	declined	NaN	42.977367	

876 rows × 76 columns

In [169…

```
nt_clean = nt_clean[(nt_clean['age']>= 18 ) & (nt_clean['age'] <= 60 ) ]
```

In [170…

```
nt_clean = nt_clean[(nt_clean['bmi_app_state'] >= 18.5 ) & (nt_clean['bmi_app_state'] <
```

In [171…

```
nt_clean.shape
```

Out[171…

```
(160834, 76)
```

In [172…

```
# import pandas_profiling

# profile = pandas_profiling.ProfileReport(nt_appRC)
# profile.to_file("nt_clean.html")
```

Dealing with missing values

- Remove household income, medical_collection_ca_2019, expected travel fl, hiv_ca, hiv_fl
- stroke_last, seizure first
- Try converting first and last fields to days since?
- current_ins_value = 0
- replacement ins = False ?
- diabetes age = -1
- alcohol weekly = mean
- remove outliers for marijuana monthly
- occupation description: 'other'
- risky_activities: 'none'
- driver's license: mode
- hiv_pos: mode
- hiv_ca and hiv_fl: drop?
- covid: mode
- previously declined: mode
- previous decline reason: 'none'
- chestpain diagnosis: 'none'
- diabetes complications: 'none'
- diabetes gestational: mode
- diabetes hemogloben: -1
- diabetes hospitalization: mode
- employment status: 'other'
- family history: mode
- final expense: 'none'
- inpatient: 'none'
- med_advice: 'none'
- med conditions: 'none'
- seizure diagnosis: 'none'
- stroke count: 0
- stroke diagnosis: 'none'

- stroke diagnosis multiselect: 'none'
- test_proc_outstanding: False
- test_proc_type: 'none'
- climbing equipment: False
- disabiliity audio_visual: False
- disability pmts reason: 'none'
- dui count: mean
- expected travel 90 days: 'none'
- expected travel: False
- expected travel muliteselect: 'none'
- mental health diagnosis: 'none'
- mental health hospitalized: False
- mental health missed work: False
- pilot student private: False
- racing 100 mph: False
- rx_increase: False
- scuba_130 ft: False
- seizure car accicent: False
- tb: False
- last worked: ?
- weight loss amount: add mean if weight losss is true, otherwise 0
    - remove outliers
- weight loss reason: 'none'
- cancer type: 'none'
- legal resident: mode
- skydive count: 0
    - remove outliers
- surgery type: 'none'
- travel countries: 'none'

Dealing with missing values

- stroke_last, seizure first
- Try converting first and last fields to days since?

- current_ins_value = 0

- replacement ins = False ?
- diabetes age = -1
- alcohol weekly = mean
- remove outliers for marijuana monthly
- occupation description: 'other'
- risky_activities: 'none'
- driver's license: mode
- hiv_pos: mode

- hiv_ca and hiv_fl: drop?
- covid: mode
- previously declined: mode
- previous decline reason: 'none'
- chestpain diagnosis: 'none'
- diabetes complications: 'none'
- diabetes gestational: mode
- diabetes hemogloben: -1
- diabetes hospitalization: mode
- employment status: 'other'
- family history: mode
- final expense: 'none'
- inpatient: 'none'
- med_advice: 'none'
- med conditions: 'none'
- seizure diagnosis: 'none'
- stroke count: 0
- stroke diagnosis: 'none'
- stroke diagnosis multiselect: 'none'
- test_proc_outstanding: False
- test_proc_type: 'none'
- climbing equipment: False
- disabiliity audio_visual: False
- disability pmts reason: 'none'
- dui count: mean
- expected travel 90 days: 'none'
- expected travel: False
- expected travel muliteselect: 'none'
- mental health diagnosis: 'none'
- mental health hospitalized: False
- mental health missed work: False
- pilot student private: False
- racing 100 mph: False
- rx_increase: False
- scuba_130 ft: False
- seizure car accicent: False
- tb: False
- last worked: ?
- weight loss amount: add mean if weight losss is true, otherwise 0
  - remove outliers
- weight loss reason: 'none'
- cancer type: 'none'
- legal resident: mode

- skydive count: 0
  - remove outliers
- surgery type: 'none'
- travel countries: 'none'

In [173…

```
nt_clean.drop(['household_income','medical_collection_ca_2019', 'hiv_ca', 'hiv_fl', 'ex
             axis=1, inplace=True)

# drop dates since it's not worth trying to clean them
nt_clean.drop(['stroke_last', 'seizure_last', 'seizure_first', 'last_worked'], axis=1,
```

- occupation description: 'other'
- risky_activities: 'none'
- previous decline reason: 'none'
- chestpain diagnosis: 'none'
- diabetes complications: 'none'
- employment status: 'other'
- final expense: 'none'
- inpatient: 'none'
- med_advice: 'none'
- med conditions: 'none'
- seizure diagnosis: 'none'
- stroke diagnosis: 'none'
- stroke diagnosis multiselect: 'none'
- test_proc_type: 'none'
- disability pmts reason: 'none'
- expected travel 90 days: 'none'
- expected travel muliteselect: 'none'
- mental health diagnosis: 'none'
- weight loss reason: 'none'
- cancer type: 'none'
- surgery type: 'none'
- travel countries: 'none'

In [174…

```
fill_values = {'occupation_description': 'other', 'risky_activities': 'none'
, 'previous_decline_reason': 'none'
, 'chestpain_diagnosis': 'none'
, 'diabetes_complications': 'none'
, 'employment_status': 'other'
, 'final_expense': 'none'
, 'inpatient': 'none'
, 'med_advice': 'none'
, 'med_conditions': 'none'
, 'seizure_diagnosis': 'none'
, 'stroke_diagnosis': 'none'
, 'stroke_diagnosis_multiselect': 'none'
, 'test_proc_type': 'none'
, 'disability_pmts_reason': 'none'
```

```
, 'expected_travel_90_days': 'none'
, 'expected_travel_multiselect': 'none'
, 'mental_health_diagnosis': 'none'
,'weight_loss_reason': 'none'
,'cancer_type': 'none'
,'surgery_type': 'none'
,'travel_countries': 'none'}

nt_clean.fillna(value = fill_values, inplace=True)
```

- driver's license: mode
- hiv_pos: mode
- covid: mode
- previously declined: mode
- diabetes gestational: mode
- diabetes hospitalization: mode
- family history: mode
- climbing equipment: False
- disabiliity audio_visual: False
- expected travel: False
- mental health hospitalized: False
- mental health missed work: False
- pilot student private: False
- racing 100 mph: False
- rx_increase: False
- scuba_130 ft: False
- seizure car accicent: False
- tb: False
- legal resident: mode

In [175…

```
# fill null values with mode in selected columns
selected_columns = [
'test_proc_outstanding',
'valid_drivers_license_app_state',
'hiv_pos',
'covid',
'previous_declined',
'diabetes_gestational',
'diabetes_hemoglobin',
'diabetes_hospitalization',
'family_history',
'climbing_equipment',
'disability_audio_visual',
'expected_travel',
'mental_health_hospitalized',
'mental_health_missed_work',
'pilot_student_private',
'racing_100mph',
'rx_increase',
'scuba_130ft',
'seizure_car_accident',
```

```
'tb',
'legal_resident',
'replacement_ins']

for col in selected_columns:
    if nt_clean[col].dtype == bool:
        nt_clean[col].fillna(nt_clean[col].mode().iloc[0], inplace=True)
    elif nt_clean[col].dtype == object:
        nt_clean[col].fillna(nt_clean[col].mode().iloc[0], inplace=True)
```

In [176…

```
num_fill_values = {'current_ins_value' : 0, 'diabetes_age' : -1, 'diabetes_hemoglobin':
                   'skydive_count': 0, 'dui_count':0}

nt_clean.fillna(value = num_fill_values, inplace=True)
```

In [177…

```
# nt_clean[['alcohol_weekly']].sort_values('alcohol_weekly', ascending = False)
nt_clean[nt_clean['alcohol_weekly'] >= 50]
```

Out[177…

	age	gender	height	weight	state	approved_risk_class	mortality_rate	bmi_app_state	in
1778	48.882592	male	64.0	160.0	WA	declined	NaN	27.460938	
7240	30.549566	female	68.0	160.0	MO	declined	NaN	24.325260	
7359	27.546082	male	68.0	170.0	GA	declined	NaN	25.845588	
7667	51.609547	male	74.0	198.0	FL	declined	NaN	25.418919	
7815	26.330452	female	64.0	218.0	OK	declined	NaN	37.415527	
...	...	...	...	...	...	...	...	...	
159400	50.109174	male	71.0	150.0	AR	declined	NaN	20.918469	
161994	50.109174	male	72.0	235.0	IN	declined	NaN	31.868248	
167135	34.971286	male	68.0	220.0	IL	declined	NaN	33.447232	
167862	27.636433	male	74.0	158.0	NC	declined	NaN	20.283784	
172285	40.249971	female	64.0	129.0	TX	declined	NaN	22.140381	

104 rows × 67 columns

In [178…

```
nt_clean[(nt_clean['alcohol_weekly'] < 50) |( nt_clean['alcohol_weekly'].isna())]
```

Out[178…

	age	gender	height	weight	state	approved_risk_class	mortality_rate	bmi_app_state	in
0	48.534878	male	68.0	195.0	TX	declined	NaN	29.646410	
1	42.716825	male	67.0	200.0	NM	declined	NaN	31.321007	
2	54.399474	male	66.0	193.0	KS	declined	NaN	31.147612	
3	50.336420	female	63.0	140.0	OK	declined	NaN	24.797178	
4	42.725039	female	61.0	132.0	IN	declined	NaN	24.938457	

	age	gender	height	weight	state	approved_risk_class	mortality_rate	bmi_app_state	in
...	...	...	...	...	...	...	...	...	
173552	35.020569	female	61.0	150.0	ME	declined	NaN	28.339156	
173553	41.312279	female	69.0	198.0	ME	declined	NaN	29.236295	
173554	40.214378	male	72.0	196.0	PA	declined	NaN	26.579475	
173555	32.887739	male	75.0	200.0	PA	declined	NaN	24.995556	
173556	37.306721	male	70.0	160.0	PA	declined	NaN	22.955102	

160730 rows × 67 columns

In [179…

```
# remove users who reported having more than 50 drinks per week
nt_clean = nt_clean[(nt_clean['alcohol_weekly'] < 50) |( nt_clean['alcohol_weekly'].isn
print(nt_clean.shape)
```

```
(160730, 67)
```

In [180…

```
# fill null values with median for alcohol weekly
nt_clean['alcohol_weekly'].fillna(value=nt_clean['alcohol_weekly'].median(), inplace=Tr
```

In [181…

```
# remove users who reported smoking marijuana more than 100 times per month
nt_clean[nt_clean['marijuana_monthly_count'] >= 100]
```

Out[181…

	age	gender	height	weight	state	approved_risk_class	mortality_rate	bmi_app_state	in
192	23.986803	female	67.0	180.0	CO	declined	NaN	28.188906	
506	29.925324	female	67.0	180.0	MO	declined	NaN	28.188906	
542	28.687790	male	68.0	138.0	LA	declined	NaN	20.980536	
566	24.559026	male	72.0	170.0	OK	declined	NaN	23.053627	
777	26.546746	female	64.0	122.0	FL	declined	NaN	20.938965	
...	...	...	...	...	...	...	...	...	
172864	27.943079	female	70.0	196.0	WA	declined	NaN	28.120000	
172909	31.778887	female	66.0	120.0	CA	declined	NaN	19.366391	
172966	30.760385	male	76.0	193.0	OR	declined	NaN	23.490132	
173101	26.251052	female	62.0	115.0	NM	declined	NaN	21.031478	
173390	32.208738	male	69.0	130.0	OK	declined	NaN	19.195547	

525 rows × 67 columns

In [182…

```
nt_clean[(nt_clean['marijuana_monthly_count'] < 100) | (nt_clean['marijuana_monthly_cou
```

Out[182…

	age	gender	height	weight	state	approved_risk_class	mortality_rate	bmi_app_state	in
0	48.534878	male	68.0	195.0	TX	declined	NaN	29.646410	
1	42.716825	male	67.0	200.0	NM	declined	NaN	31.321007	
2	54.399474	male	66.0	193.0	KS	declined	NaN	31.147612	
3	50.336420	female	63.0	140.0	OK	declined	NaN	24.797178	
4	42.725039	female	61.0	132.0	IN	declined	NaN	24.938457	
...	...	...	...	...	...	...	...	...	
173552	35.020569	female	61.0	150.0	ME	declined	NaN	28.339156	
173553	41.312279	female	69.0	198.0	ME	declined	NaN	29.236295	
173554	40.214378	male	72.0	196.0	PA	declined	NaN	26.579475	
173555	32.887739	male	75.0	200.0	PA	declined	NaN	24.995556	
173556	37.306721	male	70.0	160.0	PA	declined	NaN	22.955102	

160205 rows × 67 columns

In [183…

```
nt_clean = nt_clean[(nt_clean['marijuana_monthly_count'] < 100) | (nt_clean['marijuana_
```

In [184…

```
# fill null values with median in column 'col1'
nt_clean['marijuana_monthly_count'].fillna(value=nt_clean['marijuana_monthly_count'].me
```

In [185…

```
# check which columns still have nulls
for col in list(nt_clean.columns):
    if nt_clean[col].isnull().sum() > 0:
        print(str(col))
        print(nt_clean[col].dtype)
```

```
mortality_rate
float64
weight_loss_amount
float64
```

In [186…

```
nt_clean['replacement_ins'].isnull().sum()
```

Out[186…

```
0
```

In [187…

```
nt_clean[nt_clean['marijuana_monthly_count'].isna()]
```

Out[187…

age	gender	height	weight	state	approved_risk_class	mortality_rate	bmi_app_state	income_app_sta

0 rows × 67 columns

In [188…

```
nt_mj = nt_clean
```

In [189…

```
# fillna for users who did not lose weight
nt_mj.loc[~nt_mj['weight_loss'], 'weight_loss_amount'] = nt_mj.loc[~nt_mj['weight_loss'
```

In [190…

```
nt_clean.loc[~nt_clean['weight_loss'], 'weight_loss_amount']
```

Out[190…

```
0         0.0
1         0.0
2         0.0
4         0.0
5         0.0
         ...
173551    0.0
173553    0.0
173554    0.0
173555    0.0
173556    0.0
Name: weight_loss_amount, Length: 140853, dtype: float64
```

In [191…

```
# fillna for users who do not use marijuana
nt_mj.loc[~nt_mj['marijuana'], 'marijuana_monthly_count'] = nt_mj.loc[~nt_mj['marijuana
```

In [192…

```
nt_mj.loc[nt_mj['marijuana'] == False]['marijuana_monthly_count']
```

Out[192…

```
0         5.0
1         5.0
2         5.0
4         5.0
5         5.0
         ...
173550    5.0
173551    5.0
173553    5.0
173554    5.0
173555    5.0
Name: marijuana_monthly_count, Length: 141864, dtype: float64
```

In [193…

```
# factor = pd.factorize(nt_clean['approved_risk_class'])
# nt_clean.approved_risk_class = factor[0]
# definitions = factor[1]
# print(nt_clean.approved_risk_class.head())
# print(definitions)
```

drop mortality rate

In [194…

```
nt_clean[nt_clean['approved_risk_class'] == 'essential_plus_nt']
```

Out[194…

	age	gender	height	weight	state	approved_risk_class	mortality_rate	bmi_app_state	in
18801	49.027701	male	67.0	220.0	FL	essential_plus_nt	200.0	34.453108	
18817	33.906240	female	69.0	135.0	TX	essential_plus_nt	200.0	19.933837	
18945	48.901757	male	72.0	160.0	DC	essential_plus_nt	200.0	21.697531	
18960	41.695586	female	66.0	170.0	FL	essential_plus_nt	200.0	27.435721	
19024	48.447264	male	71.0	186.0	LA	essential_plus_nt	200.0	25.938901	
...	...	...	...	...	...	...	...	...	
173087	51.220764	female	64.0	175.0	ME	essential_plus_nt	200.0	30.035400	
173223	37.690028	male	71.0	253.0	GA	essential_plus_nt	200.0	35.282484	
173276	41.462864	female	65.0	130.0	NE	essential_plus_nt	200.0	21.630769	
173403	44.652525	male	67.0	165.0	FL	essential_plus_nt	200.0	25.839831	
173472	40.600423	female	64.0	160.0	FL	essential_plus_nt	200.0	27.460938	

1460 rows × 67 columns

In [197…

```
nt_appRC = nt_clean.drop('mortality_rate', axis=1)
nt_appRC = nt_appRC[nt_appRC['approved_risk_class'] != 'essential_plus_nt']
nt_appRC.groupby('approved_risk_class').count()
```

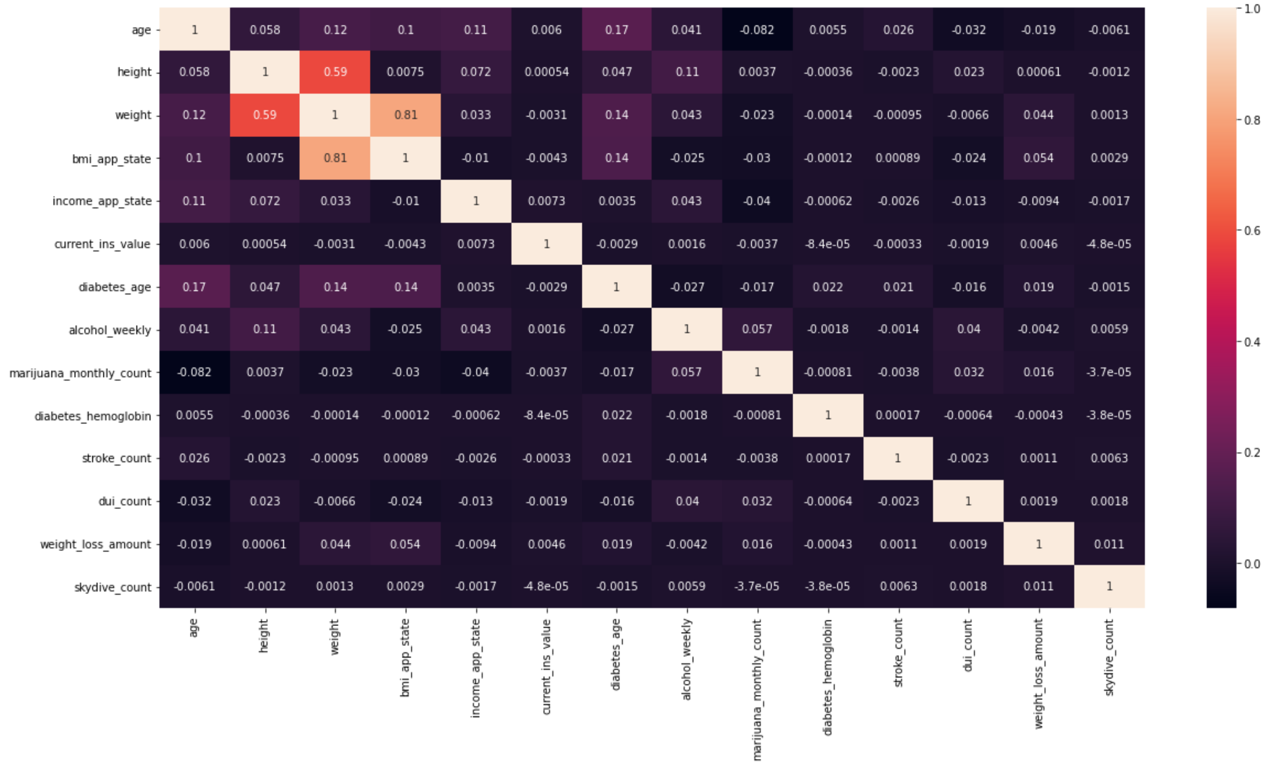
Out[197…

	age	gender	height	weight	state	bmi_app_state	income_app_state	current_ins
approved_risk_class								
declined	94513	94513	94513	94513	94513	94513	94513	94513
elite_nt	8931	8931	8931	8931	8931	8931	8931	8931
essential_nt	21681	21681	21681	21681	21681	21681	21681	21681
preferred_nt	10502	10502	10502	10502	10502	10502	10502	10502
select_nt	23118	23118	23118	23118	23118	23118	23118	23118

5 rows × 65 columns

In [196…

```
df_num = nt_appRC.select_dtypes(include='number')
corr_mat = df_num.corr()

import seaborn as sn
import matplotlib.pyplot as plt
plt.rcParams['figure.figsize'] = (20, 10)
sn.heatmap(corr_mat, annot=True)
plt.show()
```



In [117…

```
import pandas_profiling
# import ydata_profiling

profile = pandas_profiling.ProfileReport(nt_appRC)
profile.to_file("nt_clean.html")
```

```
C:\Users\milan\AppData\Local\Temp/ipykernel_40260/2879794622.py:1: DeprecationWarning: `
import pandas_profiling` is going to be deprecated by April 1st. Please use `import ydat
a_profiling` instead.
  import pandas_profiling
```

In [119…

```
# transform categorical features

from category_encoders import *

encoder = TargetEncoder()
factor = pd.factorize(nt_appRC['approved_risk_class'])
y = factor[0]
definitions = factor[1]

X = nt_appRC.drop(columns = 'approved_risk_class')
enc = TargetEncoder(min_samples_leaf=20, smoothing=10).fit(X,y)
nt_transformed = enc.transform(X)
nt_transformed
```

```
C:\Users\milan\anaconda3\lib\site-packages\category_encoders\target_encoder.py:122: Futu
reWarning: Default parameter min_samples_leaf will change in version 2.6.See https://git
hub.com/scikit-learn-contrib/category_encoders/issues/327
  warnings.warn("Default parameter min_samples_leaf will change in version 2.6."
C:\Users\milan\anaconda3\lib\site-packages\category_encoders\target_encoder.py:127: Futu
```

```
reWarning: Default parameter smoothing will change in version 2.6.See https://github.co
m/scikit-learn-contrib/category_encoders/issues/327
  warnings.warn("Default parameter smoothing will change in version 2.6."
```

Out[119…

	age	gender	height	weight	state	bmi_app_state	income_app_state	current_ins	cu
0	48.534878	1.029396	68.0	195.0	1.140601	29.646410	36400	1.015067	
1	42.716825	1.029396	67.0	200.0	0.998968	31.321007	9528	1.015067	
2	54.399474	1.029396	66.0	193.0	0.888889	31.147612	9396	1.015067	
3	50.336420	1.014664	63.0	140.0	0.905008	24.797178	82000	1.015067	
4	42.725039	1.014664	61.0	132.0	0.751253	24.938457	72000	1.015067	
...	...	...	...	...	...	...	...	...	
173552	35.020569	1.014664	61.0	150.0	0.985413	28.339156	70000	1.015067	
173553	41.312279	1.014664	69.0	198.0	0.985413	29.236295	169000	1.015067	
173554	40.214378	1.029396	72.0	196.0	0.941686	26.579475	11772	1.015067	
173555	32.887739	1.029396	75.0	200.0	0.941686	24.995556	369200	1.015067	
173556	37.306721	1.029396	70.0	160.0	0.941686	22.955102	36400	1.015067	

158745 rows × 65 columns

In [120…

```
print(definitions)
```

```
Index(['declined', 'elite_nt', 'select_nt', 'essential_nt', 'preferred_nt'], dtype='obje
ct')
```

In [121…

```
nt_transformed['approved_risk_class'] = y
nt_transformed.head()
```

Out[121…

	age	gender	height	weight	state	bmi_app_state	income_app_state	current_ins	current_
0	48.534878	1.029396	68.0	195.0	1.140601	29.646410	36400	1.015067	
1	42.716825	1.029396	67.0	200.0	0.998968	31.321007	9528	1.015067	
2	54.399474	1.029396	66.0	193.0	0.888889	31.147612	9396	1.015067	
3	50.336420	1.014664	63.0	140.0	0.905008	24.797178	82000	1.015067	
4	42.725039	1.014664	61.0	132.0	0.751253	24.938457	72000	1.015067	

5 rows × 66 columns

In [122…

```
nt_transformed.shape
```

Out[122…

```
(158745, 66)
```

In [123…

```
# save cleaned and transformed dataframe
nt_transformed.to_csv(r'C:\Users\milan\Documents\GitHub\DSC-680\approved_risk_class_cle
```

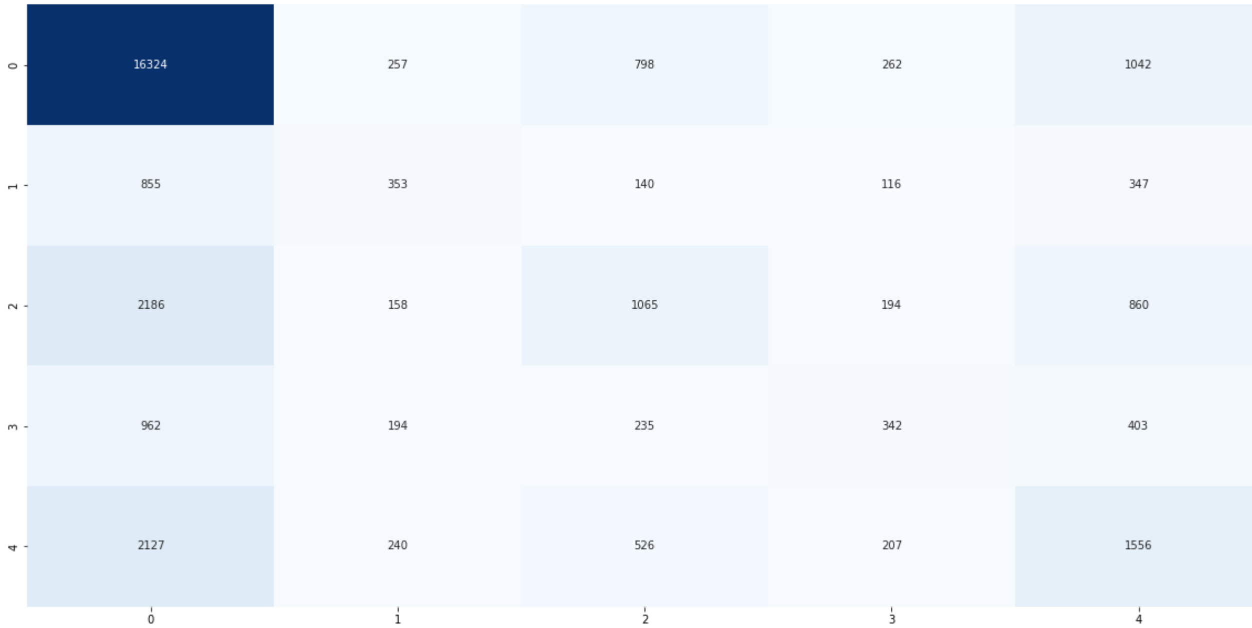
In [107…

```
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split

df = pd.read_csv(r'C:\Users\milan\Documents\GitHub\DSC-680\approved_risk_class_clean.cs

y = df['approved_risk_class']
X = df.drop('approved_risk_class', axis=1)


# Split data into training and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=4

# # Create random forest classifier with 100 trees
clf = RandomForestClassifier(n_estimators=100, random_state=42)

# # Train the classifier on the training set
clf.fit(X_train, y_train)

# # Make predictions on the test set
y_pred = clf.predict(X_test)

# # Evaluate the performance of the classifier
accuracy = clf.score(X_test, y_test)
print("Accuracy:", accuracy)
```

```
Accuracy: 0.6186021606979747
```

In [108…

```
import matplotlib.pyplot as plt
import seaborn as sn
from sklearn.metrics import confusion_matrix
matrix = confusion_matrix(y_test.values, y_pred)
df = pd.DataFrame(matrix)
sn.heatmap(df, annot = True, cbar = None, cmap = "Blues", fmt='g')
plt.show()

# Note: rows are true values, columns are predicted
# Rows = recall
# Columns = precision
```



In [109…

```
from sklearn.metrics import classification_report
print(classification_report(y_test.values, y_pred))
```

```
              precision    recall  f1-score   support

    declined       0.73      0.87      0.79     18683
    elite_nt       0.29      0.19      0.23      1811
essential_nt       0.39      0.24      0.29      4463
preferred_nt       0.31      0.16      0.21      2136
   select_nt       0.37      0.33      0.35      4656

    accuracy                           0.62     31749
   macro avg       0.42      0.36      0.38     31749
weighted avg       0.57      0.62      0.59     31749
```

In [ ]:

```
# there are only 295 instances of essential_plus_nt, so I will remove this risk class a
```

In [124…

```
nt_transformed.head()
```

Out[124…

	age	gender	height	weight	state	bmi_app_state	income_app_state	current_ins	current_
0	48.534878	1.029396	68.0	195.0	1.140601	29.646410	36400	1.015067	
1	42.716825	1.029396	67.0	200.0	0.998968	31.321007	9528	1.015067	
2	54.399474	1.029396	66.0	193.0	0.888889	31.147612	9396	1.015067	
3	50.336420	1.014664	63.0	140.0	0.905008	24.797178	82000	1.015067	
4	42.725039	1.014664	61.0	132.0	0.751253	24.938457	72000	1.015067	

5 rows × 66 columns

In [129…

```
train, test = train_test_split(nt_transformed, test_size=0.2, random_state=42)

features = train.drop('approved_risk_class', axis=1)
target = train['approved_risk_class']
test_features = test.drop('approved_risk_class', axis=1)
test_target = test['approved_risk_class']
```

In [130…

```
# Create pipeline
from sklearn.ensemble import RandomForestClassifier
from sklearn import metrics
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
from sklearn.inspection import permutation_importance
from sklearn.preprocessing import StandardScaler
from sklearn.pipeline import Pipeline, FeatureUnion

pipe = Pipeline([
    ('scaler', StandardScaler()),
    ('rf', RandomForestClassifier())
])
```

In [131…

```
# Set Parameters
params = {
    'rf__n_estimators': [120, 140],
    'rf__max_depth': [30, 50],
    'rf__min_samples_split': [2,3],
    'rf__min_samples_leaf': [3,5]
#     'rf__class_weight': [{0: 1, 1: 1}, {0: 1, 1:5},{0:1,1:3}, 'balanced']

}
```

In [137…

```
# GridSearch, Fit, Score
from sklearn.model_selection import GridSearchCV, TimeSeriesSplit, train_test_split
RF_gs = GridSearchCV(pipe, param_grid=params ,scoring = 'f1_weighted', cv = 3)
RF_gs.fit(features, target)
```

Out[137…

```
GridSearchCV(cv=3,
             estimator=Pipeline(steps=[('scaler', StandardScaler()),
                                       ('rf', RandomForestClassifier())]),
             param_grid={'rf__max_depth': [30, 50],
                         'rf__min_samples_leaf': [3, 5],
                         'rf__min_samples_split': [2, 3],
                         'rf__n_estimators': [120, 140]},
             scoring='f1_weighted')
```

In [138…

```
RF_gs.best_estimator_
```

Out[138…

```
Pipeline(steps=[('scaler', StandardScaler()),
                ('rf',
                 RandomForestClassifier(max_depth=50, min_samples_leaf=3,
                                        min_samples_split=3,
                                        n_estimators=120))])
```

In [139…

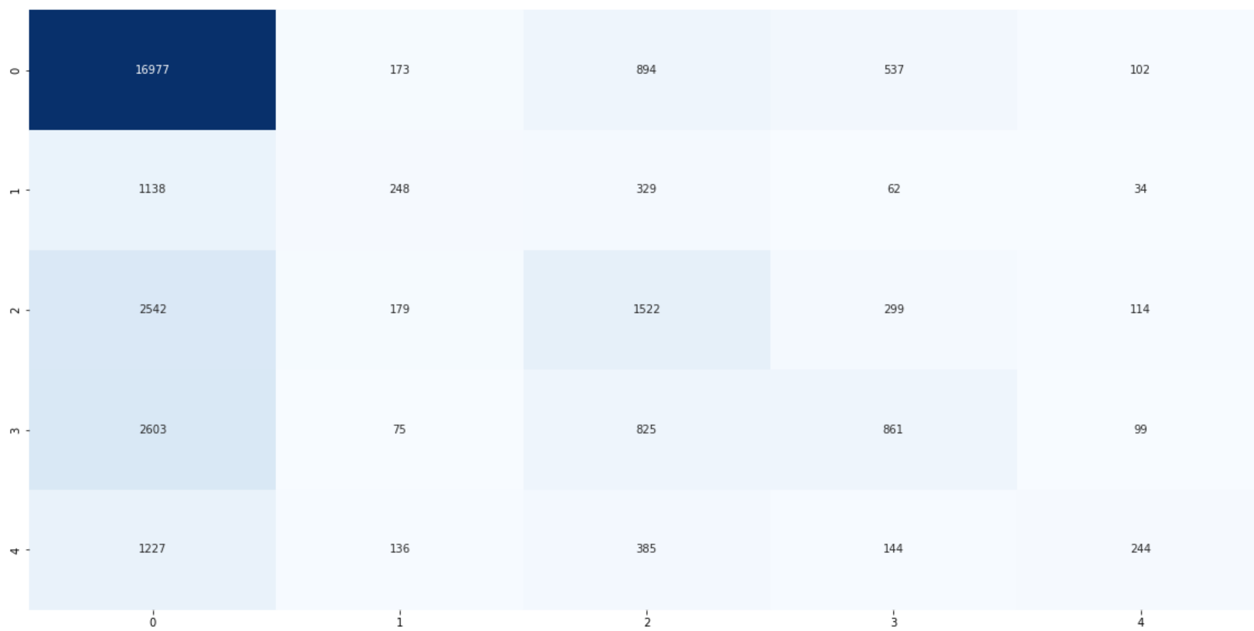
```
pipe = Pipeline([("standardizer", StandardScaler()), ("rf", RandomForestClassifier(max_
                                                        min_samples_leaf=3,
                                                        n_estimators=120))])
pipe.fit(features, target)
rf_preds = pipe.predict(test_features)
accuracy_score(rf_preds, test_target)
```

Out[139…

```
0.62527953636335
```

In [140…

```
matrix = confusion_matrix(test_target, rf_preds)
df = pd.DataFrame(matrix)
sn.heatmap(df, annot = True, cbar = None, cmap = "Blues", fmt='g')
plt.show()
```



In [141…

```
from sklearn.metrics import classification_report
print(classification_report(test_target, rf_preds))
```

```
              precision    recall  f1-score   support

           0       0.69      0.91      0.79     18683
           1       0.31      0.14      0.19      1811
           2       0.38      0.33      0.35      4656
           3       0.45      0.19      0.27      4463
           4       0.41      0.11      0.18      2136

    accuracy                           0.63     31749
   macro avg       0.45      0.34      0.36     31749
weighted avg       0.57      0.63      0.58     31749
```

In [ ]:

```
```

In [135…

```
import sklearn
sorted(sklearn.metrics.SCORERS.keys())
```

Out[135…

```
['accuracy',
 'adjusted_mutual_info_score',
 'adjusted_rand_score',
 'average_precision',
 'balanced_accuracy',
 'completeness_score',
 'explained_variance',
 'f1',
 'f1_macro',
 'f1_micro',
 'f1_samples',
 'f1_weighted',
 'fowlkes_mallows_score',
 'homogeneity_score',
 'jaccard',
 'jaccard_macro',
 'jaccard_micro',
 'jaccard_samples',
 'jaccard_weighted',
 'max_error',
 'mutual_info_score',
 'neg_brier_score',
 'neg_log_loss',
 'neg_mean_absolute_error',
 'neg_mean_absolute_percentage_error',
 'neg_mean_gamma_deviance',
 'neg_mean_poisson_deviance',
 'neg_mean_squared_error',
 'neg_mean_squared_log_error',
 'neg_median_absolute_error',
 'neg_root_mean_squared_error',
 'normalized_mutual_info_score',
 'precision',
 'precision_macro',
 'precision_micro',
 'precision_samples',
 'precision_weighted',
 'r2',
 'rand_score',
 'recall',
 'recall_macro',
 'recall_micro',
 'recall_samples',
 'recall_weighted',
 'roc_auc',
 'roc_auc_ovo',
 'roc_auc_ovo_weighted',
 'roc_auc_ovr',
 'roc_auc_ovr_weighted',
 'top_k_accuracy',
 'v_measure_score']
```

In [ ]: