Incident Impact Predictions

1. Import necessary libraries

In [1]:
```
import pandas as pd
import numpy as np

import matplotlib.pyplot as plt
import seaborn as sns

import datetime
from datetime import datetime as dt

from sklearn.preprocessing import OrdinalEncoder, LabelEncoder, OneHotEncoc
from sklearn.model_selection import train_test_split


from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier, AdaBoostClassifier
from xgboost import XGBClassifier
from sklearn.svm import SVC
from sklearn.metrics import confusion_matrix, ConfusionMatrixDisplay, plot_

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

In [2]:
```
pd.set_option('max_rows', 100)
pd.set_option('max_columns', None)
```

In [3]:
```
#sns.set_context("notebook", font_scale=0.8)
#%matplotlib notebook
```

2.Load data

In [4]:
```
incident_df = pd.read_csv('incident_event_log.csv')
```

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

Out[5]:

	number	incident_state	active	reassignment_count	reopen_count	sys_mod_count	made_sl
0	INC0000045	New	True	0	0	0	Tru
1	INC0000045	Resolved	True	0	0	2	Tru
2	INC0000045	Resolved	True	0	0	3	Tru

	number	incident_state	active	reassignment_count	reopen_count	sys_mod_count	made_sl
3	INC0000045	Closed	False	0	0	4	Tru

2.1 Data understanding

In [6]:
```
incident_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 141712 entries, 0 to 141711
Data columns (total 36 columns):
 #   Column                   Non-Null Count   Dtype
---  ------                   --------------   -----
 0   number                   141712 non-null  object
 1   incident_state           141712 non-null  object
 2   active                   141712 non-null  bool
 3   reassignment_count       141712 non-null  int64
 4   reopen_count             141712 non-null  int64
 5   sys_mod_count            141712 non-null  int64
 6   made_sla                 141712 non-null  bool
 7   caller_id                141712 non-null  object
 8   opened_by                141712 non-null  object
 9   opened_at                141712 non-null  object
 10  sys_created_by           141712 non-null  object
 11  sys_created_at           141712 non-null  object
 12  sys_updated_by           141712 non-null  object
 13  sys_updated_at           141712 non-null  object
 14  contact_type             141712 non-null  object
 15  location                 141712 non-null  object
 16  category                 141712 non-null  object
 17  subcategory              141712 non-null  object
 18  u_symptom                141712 non-null  object
 19  cmdb_ci                  141712 non-null  object
 20  impact                   141712 non-null  object
 21  urgency                  141712 non-null  object
 22  priority                 141712 non-null  object
 23  assignment_group         141712 non-null  object
 24  assigned_to              141712 non-null  object
 25  knowledge                141712 non-null  bool
 26  u_priority_confirmation  141712 non-null  bool
 27  notify                   141712 non-null  object
 28  problem_id               141712 non-null  object
 29  rfc                      141712 non-null  object
 30  vendor                   141712 non-null  object
 31  caused_by                141712 non-null  object
 32  closed_code              141712 non-null  object
 33  resolved_by              141712 non-null  object
 34  resolved_at              141712 non-null  object
 35  closed_at                141712 non-null  object
dtypes: bool(4), int64(3), object(29)
memory usage: 35.1+ MB
```

In [7]:
```
incident_df1 = incident_df.copy() # Copy of original dataframe for EDA

# Replace all '?' with np.nan
incident_df1 = incident_df1.replace(to_replace='?', value=np.nan)

# Null values by column:
null_val_sumry = pd.DataFrame(incident_df1.isna().sum()).reset_index()
null_val_sumry.columns = ['col_name', 'null_vals']
null_val_sumry['perc_null_vals'] = null_val_sumry['null_vals'].apply(lambda
```

In [8]:
```
null_val_cols = null_val_sumry[null_val_sumry['perc_null_vals']>0]
null_val_cols
```

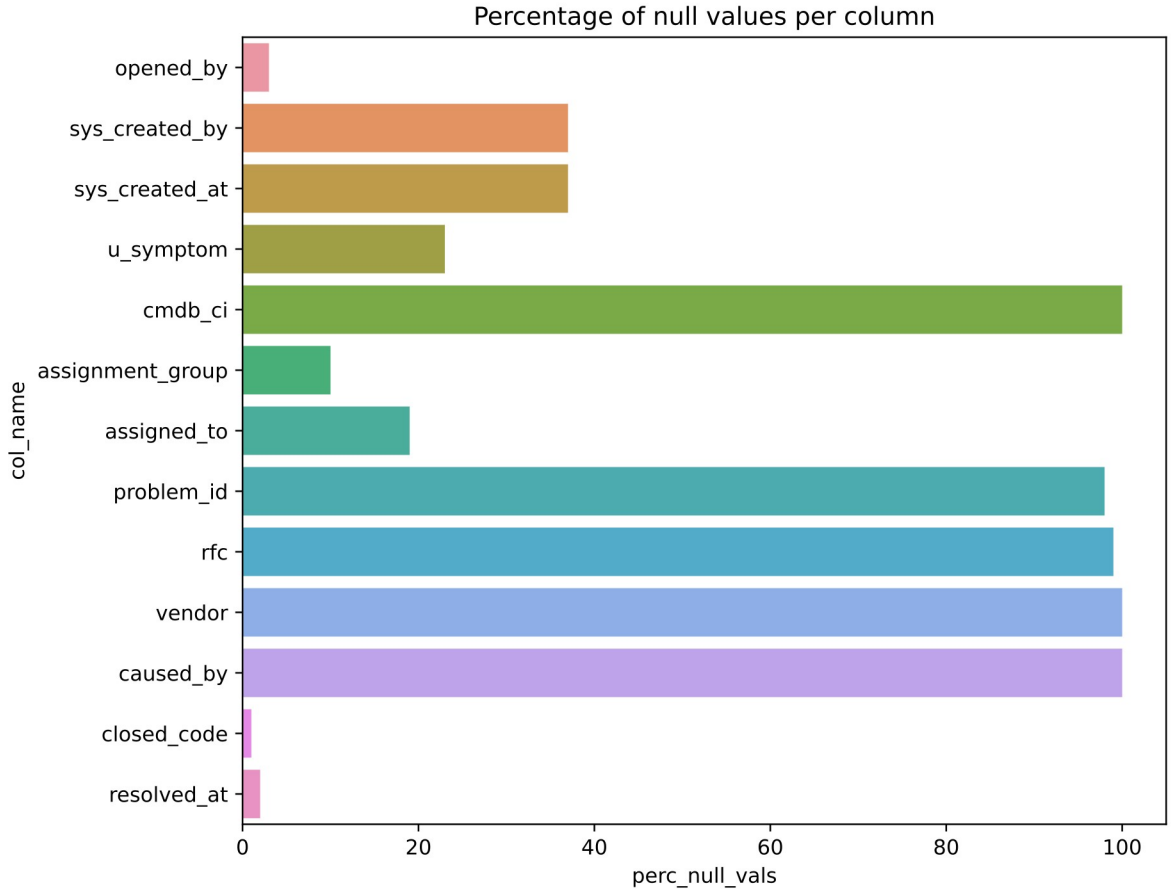
Out[8]:

	col_name	null_vals	perc_null_vals
8	opened_by	4835	3
10	sys_created_by	53076	37
11	sys_created_at	53076	37
18	u_symptom	32964	23
19	cmdb_ci	141267	100
23	assignment_group	14213	10
24	assigned_to	27496	19
28	problem_id	139417	98
29	rfc	140721	99
30	vendor	141468	100
31	caused_by	141689	100
32	closed_code	714	1
34	resolved_at	3141	2

In [9]:
```
# Total null values.
incident_df1.isna().sum().sum()
```

Out[9]:
```
894597
```

In [10]:
```
# Visualizing the null values.
fig, ax = plt.subplots(figsize=(8,7))
sns.barplot(y='col_name', x='perc_null_vals', data=null_val_cols, ax=ax)
plt.title('Percentage of null values per column')
plt.show()
```



In [11]:

```
# cols = incident_df1.columns
# colours = ['#000099', '#ffff00'] # specify the colours - yellow is missir
# fig, ax = plt.subplots(figsize=(8,6))
# sns.heatmap(incident_df1[cols].isnull(),cmap=sns.color_palette(colours),
# plt.show()

# Note: This cell crashes in normal mode but works when we set %matplotlib
```

2.1.1 Observations

- 141712 and 36 Data columns.
- Mixture of categorical, numeric and timestamp(recorded as object)
- null values recorded as '?' and '-100'.

3. EDA

3.1 Dataset details

In [12]:
```
col_list1 = incident_df1.columns

# Count the no of numeric columns.
numeric = [col for col in col_list1 if (incident_df1[col].dtypes == 'int64'

# boolean columns
bool_cols = ['active', 'made_sla', 'knowledge', 'u_priority_confirmation',

# Count the categorical columns.
categorical_all = [col for col in col_list1 if incident_df1[col].dtypes !=

## list of datetime columns.
datetime_all = ['opened_at', 'sys_created_at', 'sys_updated_at', 'resolved_

categorical_filt = [col for col in categorical_all if col not in datetime_a
cat1 = [col for col in categorical_filt if col not in bool_cols]

columns_summary = {'Numeric columns':len(numeric),
                   'Boolean columns':len(bool_cols),
                   'Categorical columns':len(cat1),
                   'Datetime columns': len(datetime_all)}
```

In [13]:
```
#unique_val_dict
columns_summary_df = pd.concat([pd.Series(columns_summary.keys()),
                                pd.Series(columns_summary.values())],
                                axis=1)
columns_summary_df.columns = ['column type', 'No. of columns']
```

In [14]:
```
incident_df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 141712 entries, 0 to 141711
Data columns (total 36 columns):
 #   Column                  Non-Null Count   Dtype
---  ------                  --------------   -----
 0   number                  141712 non-null  object
 1   incident_state          141712 non-null  object
 2   active                  141712 non-null  bool
 3   reassignment_count      141712 non-null  int64
 4   reopen_count            141712 non-null  int64
 5   sys_mod_count           141712 non-null  int64
 6   made_sla                141712 non-null  bool
 7   caller_id               141683 non-null  object
 8   opened_by               136877 non-null  object
 9   opened_at               141712 non-null  object
 10  sys_created_by          88636 non-null   object
 11  sys_created_at          88636 non-null   object
 12  sys_updated_by          141712 non-null  object
 13  sys_updated_at          141712 non-null  object
 14  contact_type            141712 non-null  object
 15  location                141636 non-null  object
 16  category                141634 non-null  object
 17  subcategory             141601 non-null  object
 18  u_symptom               108748 non-null  object
 19  cmdb_ci                 445 non-null     object
 20  impact                  141712 non-null  object
 21  urgency                 141712 non-null  object
 22  priority                141712 non-null  object
 23  assignment_group        127499 non-null  object
```

```
 24  assigned_to              114216 non-null  object
 25  knowledge                141712 non-null  bool
 26  u_priority_confirmation  141712 non-null  bool
 27  notify                   141712 non-null  object
 28  problem_id               2295 non-null    object
 29  rfc                      991 non-null     object
 30  vendor                   244 non-null     object
 31  caused_by                23 non-null      object
 32  closed_code              140998 non-null  object
 33  resolved_by              141486 non-null  object
 34  resolved_at              138571 non-null  object
 35  closed_at                141712 non-null  object
dtypes: bool(4), int64(3), object(29)
memory usage: 35.1+ MB
```

In [15]:
```
columns_summary_df
```

Out[15]:

	column type	No. of columns
0	Numeric columns	3
1	Boolean columns	5
2	Categorical columns	23
3	Datetime columns	5

3.2 Data preprocessing | Visualizations

3.2.1 Handling null values

In [16]:
```
# Function to remove all null columns(EDA)
def null_value_remove(Xdf):
    # Replace all '?' with np.nan
    Xdf = Xdf.replace(to_replace='?', value=np.nan)
    Xdf['incident_state'] = Xdf['incident_state'].replace('-100', np.nan)

    # Delete all columns with more than 20% null values.
    del_cols = ['cmdb_ci', 'problem_id', 'rfc', 'vendor', 'caused_by', 'sys
    Xdf = Xdf.drop(del_cols, axis=1)

    return Xdf
```

In [17]:
```
# Function to impute with mode values (EDA/Feature engineering)
def null_value_impute(Xdf):
    Xdf['caller_id'] = Xdf['caller_id'].fillna(value=Xdf['caller_id'].mode(
    Xdf['opened_by'] = Xdf['opened_by'].fillna(value=Xdf['opened_by'].mode(
    Xdf['location'] = Xdf['location'].fillna(value=Xdf['location'].mode()[0
    Xdf['category'] = Xdf['category'].fillna(value=Xdf['category'].mode()[0
    Xdf['subcategory']= Xdf['subcategory'].fillna(value=Xdf['subcategory'].
    Xdf['u_symptom'] = Xdf['u_symptom'].fillna(value=Xdf['u_symptom'].mode(
    Xdf['assignment_group'] = Xdf['assignment_group'].fillna(value=Xdf['ass
    Xdf['assigned_to'] = Xdf['assigned_to'].fillna(value=Xdf['assigned_to']
    Xdf['resolved_by'] = Xdf['resolved_by'].fillna(value=Xdf['resolved_by']

    # Wherever there are null values, pad them(substitute them) with the pr
    Xdf['closed_code'] = Xdf['closed_code'].fillna(method='pad')

    # Replace the null values in 'incident_state' with most frequent value
    #replace_val = 'Active' #Xdf['incident_state'].mode()[0] - hardcoded fo
    Xdf['incident_state'] = Xdf['incident_state'].fillna(value=Xdf['incider

    return Xdf
```

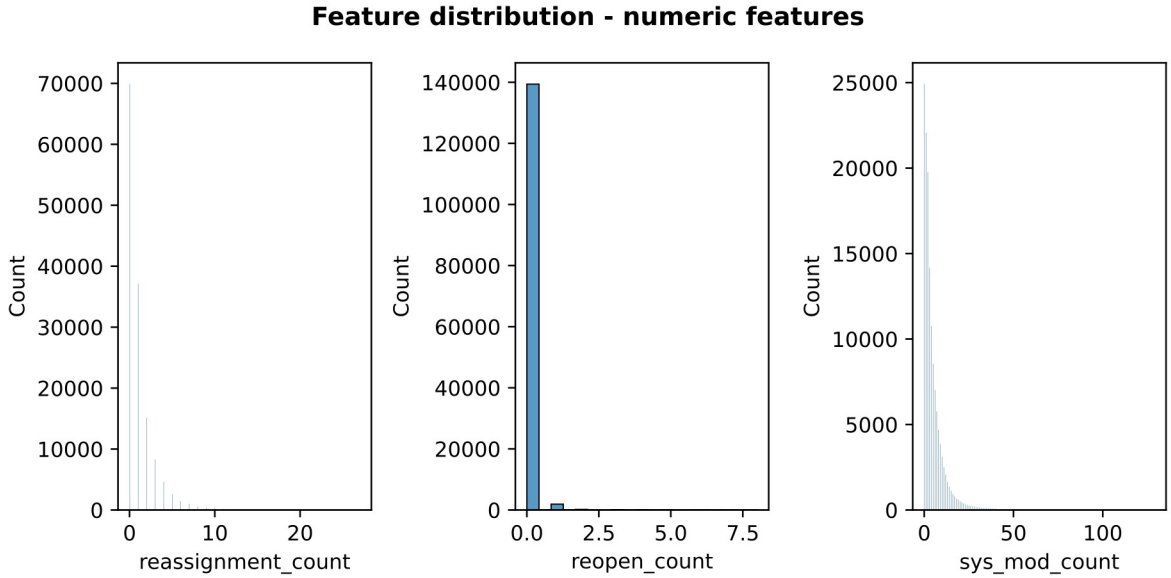
3.2.2 Encoding boolean features

In [18]:
```
# Function to encode the boolean features (EDA/model building)
def boolean_encoder(Xdf):
    Xdf['active'] = Xdf['active'].apply(lambda x:1 if x==True else 0)
    Xdf['made_sla'] = Xdf['made_sla'].apply(lambda x:1 if x==True else 0)
    Xdf['knowledge'] = Xdf['knowledge'].apply(lambda x:1 if x==True else 0)
    Xdf['u_priority_confirmation'] = Xdf['u_priority_confirmation'].apply(l
    Xdf['notify'] = Xdf['notify'].apply(lambda x:1 if x=='Send Email' else

    return Xdf
```

In [19]:
```
incident_df1 = null_value_remove(incident_df1)
incident_df1 = null_value_impute(incident_df1)
incident_df1 = boolean_encoder(incident_df1)
```

3.2.3 Visualizing numeric columns

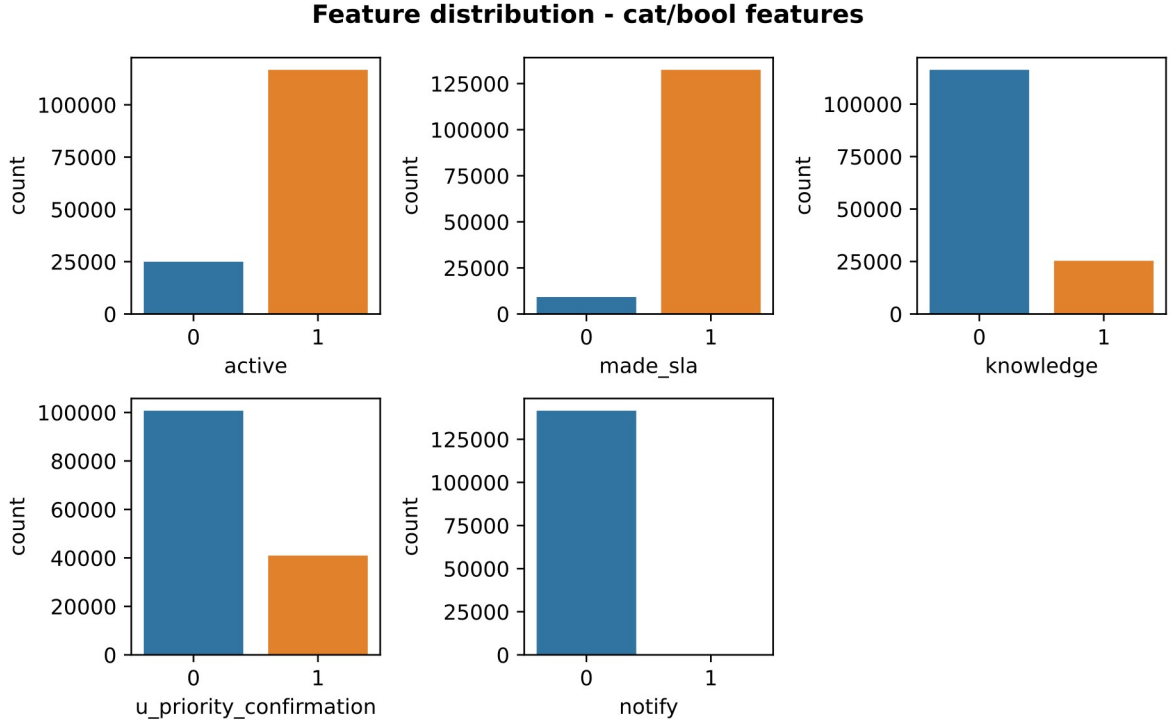
In [20]:
```
fig, axes = plt.subplots(1, 3 , figsize=(8,4))#
axes = axes.flatten()

for idx, ax in enumerate(axes):
    sns.histplot(data=incident_df1, x=incident_df1[numeric[idx]], ax=ax)

fig.suptitle('Feature distribution - numeric features', ha='center', fontwe
fig.tight_layout()
plt.show()
```



3.2.4 Visualizing boolean columns

In [21]:

```
fig, axes = plt.subplots(2,3, figsize=(8,5)) #figsize=(4,4)
fig.delaxes(axes[1,2])
axes = axes.flatten()

for idx, ax in enumerate(axes):
    if idx < 5:
        sns.countplot(data=incident_df1, x=incident_df1[bool_cols[idx]], ax

fig.suptitle('Feature distribution - cat/bool features', ha='center', fontw
fig.tight_layout()
plt.show()
```



3.2.5 understanding categorical features

In [22]:
```
datetime_cols = ['opened_at', 'sys_updated_at', 'resolved_at', 'closed_at']
# Note: 'sys_created_at' was deleted due to presence of null vals
```

In [23]:
```
datetime_cols = ['opened_at', 'sys_updated_at', 'resolved_at', 'closed_at']
# Note: 'sys_created_at' was deleted due to presence of null vals
num_bool_dt = numeric + bool_cols + datetime_cols
cat_cols = [col for col in incident_df1.columns if col not in num_bool_dt]
```

In [24]:
```
unique_val_dict = dict.fromkeys(cat_cols)
for key in cat_cols:
    unique_val_dict[key] = len(incident_df1[key].unique())
```

In [25]:
```
#unique_val_dict
unique_df = pd.concat([pd.Series(unique_val_dict.keys()),
                       pd.Series(unique_val_dict.values())],
                       axis=1)
unique_df.columns = ['columns', 'num_unique_vals']
```

In [26]:
```
unique_df
```

Out[26]:

	columns	num_unique_vals
0	number	24918
1	incident_state	8
2	caller_id	5244
3	opened_by	207
4	sys_updated_by	846
5	contact_type	5
6	location	224
7	category	58
8	subcategory	254
9	u_symptom	525
10	impact	3
11	urgency	3
12	priority	4
13	assignment_group	78
14	assigned_to	234
15	closed_code	17
16	resolved_by	216

In [27]:
```
# All cols with less than 100 unique values will be visualized.
unique_df[unique_df['num_unique_vals']< 100]
```

Out[27]:

	columns	num_unique_vals
1	incident_state	8
5	contact_type	5
7	category	58
10	impact	3
11	urgency	3
12	priority	4
13	assignment_group	78
15	closed_code	17

In [28]:

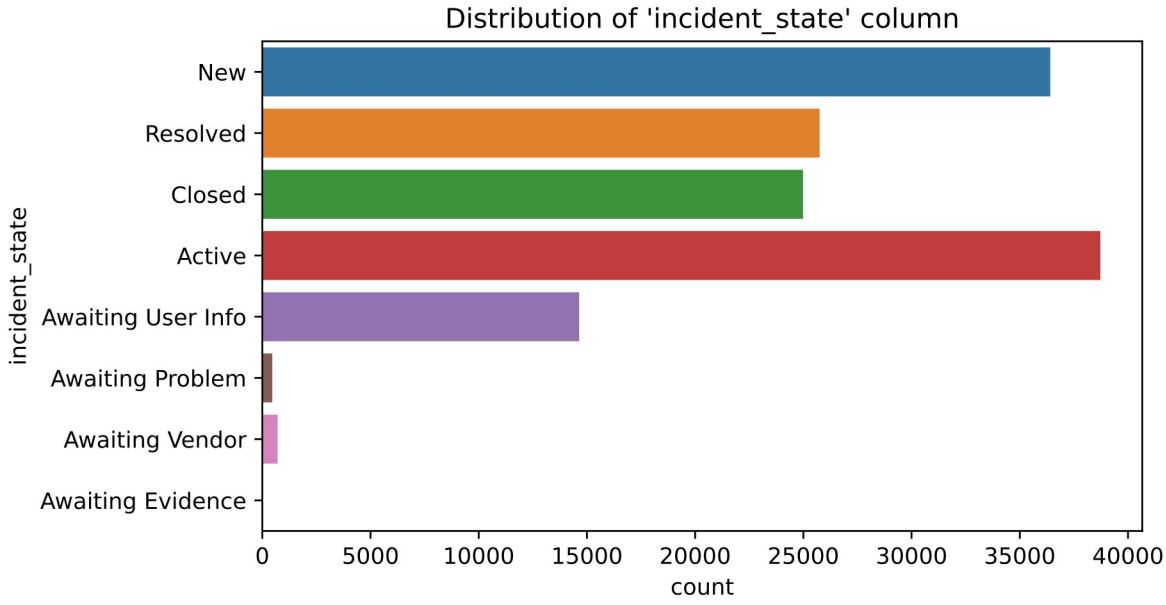
```
# All cols with more than 100 unique values - Very high cardinality feature
unique_df[unique_df['num_unique_vals'] > 100]
```

Out[28]:

	columns	num_unique_vals
0	number	24918
2	caller_id	5244
3	opened_by	207
4	sys_updated_by	846
6	location	224
8	subcategory	254
9	u_symptom	525
14	assigned_to	234
16	resolved_by	216

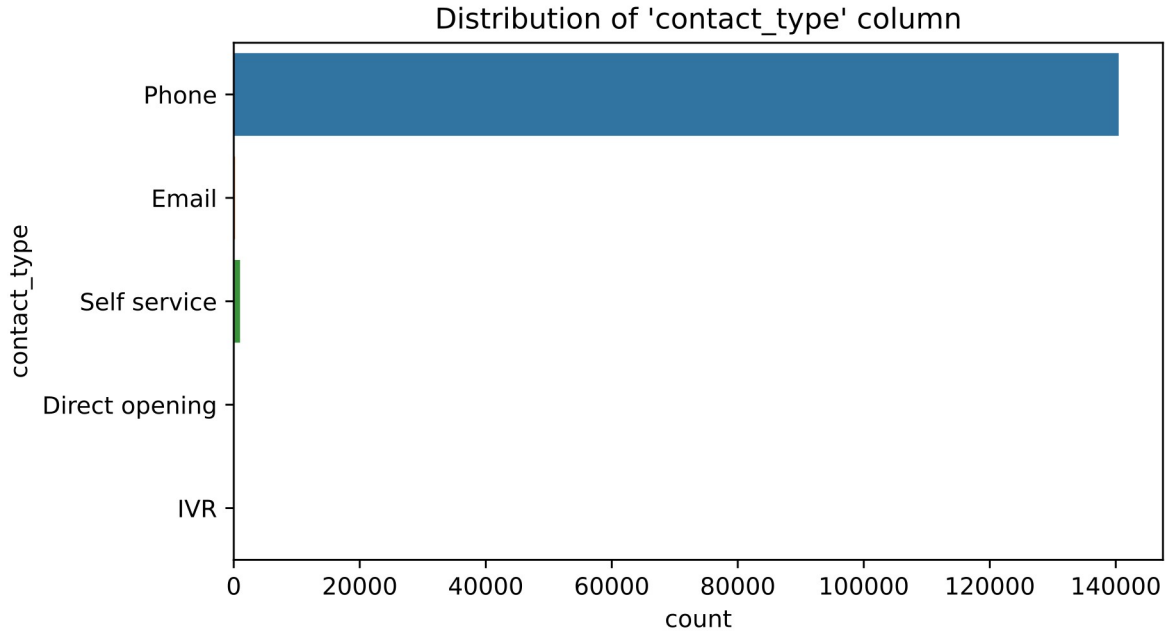
In [29]:

```
fig, ax = plt.subplots()
sns.countplot(data=incident_df1, y=incident_df1['incident_state'], ax=ax)
plt.title("Distribution of 'incident_state' column")
plt.show()
```



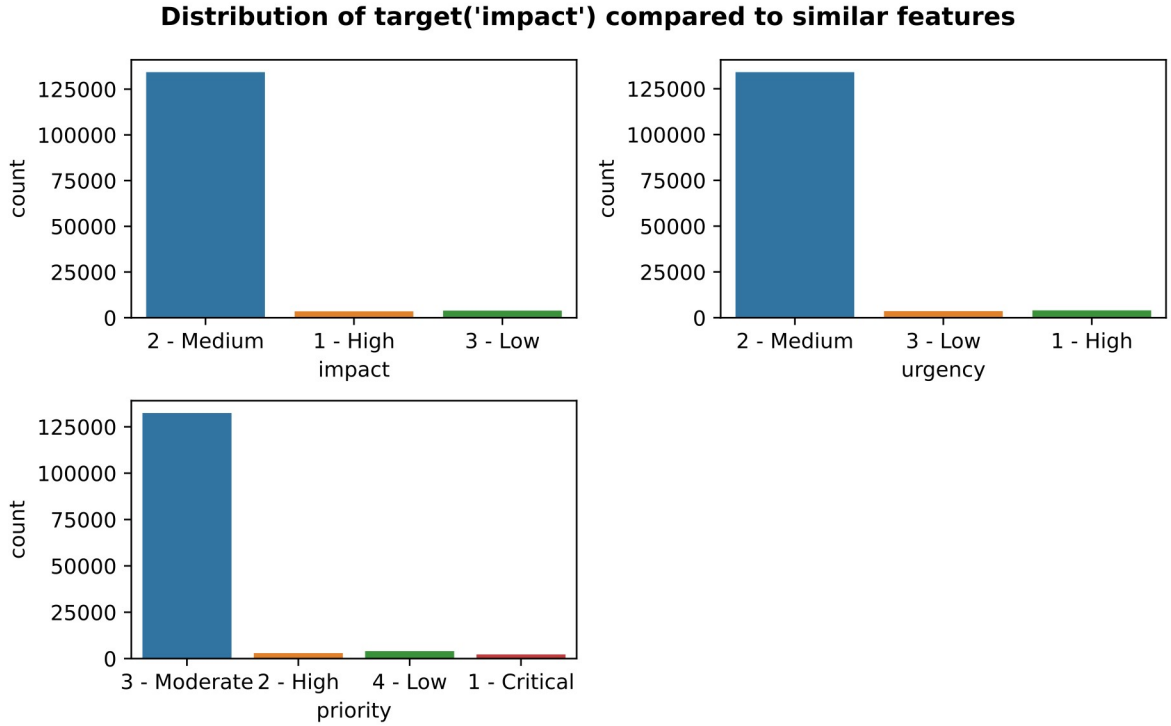
In [30]:

```
fig, ax = plt.subplots()
sns.countplot(data=incident_df1, y=incident_df1['contact_type'], ax=ax)
plt.title("Distribution of 'contact_type' column")
plt.show()
```



In [31]:

```
fig, ax = plt.subplots(2,2, figsize=(8,5))
fig.delaxes(ax[1,1])
ax = ax.flatten()

sns.countplot(data=incident_df1, x=incident_df1['impact'], ax=ax[0])
sns.countplot(data=incident_df1, x=incident_df1['urgency'], ax=ax[1])
sns.countplot(data=incident_df1, x=incident_df1['priority'], ax=ax[2])

fig.suptitle("Distribution of target('impact') compared to similar features
fig.tight_layout()
plt.show()
```



In [32]:
```
# Summary statistics for categorical columns.
incident_df1[cat_cols].describe(include=['O'])
```

Out[32]:

	number	incident_state	caller_id	opened_by	sys_updated_by	contact_type	location
count	141712	141712	141712	141712	141712	141712	141712
unique	24918	8	5244	207	846	5	224
top	INC0019396	Active	Caller 1904	Opened by 17	Updated by 908	Phone	Location 204
freq	58	38721	1454	46301	36162	140462	31766

3.2.6 Encoding categorical features

In [33]:
```
# Function to encode categorical columns with high cardinality (EDA/Feature
def categorical_encoder(Xdf):
    # scale:
    urgency_scale = {'1 - High':1, '2 - Medium':2, '3 - Low':3}
    priority_scale = {'1 - Critical':1, '2 - High':2, '3 - Moderate':3, '4

    Xdf['urgency'] = Xdf['urgency'].map(urgency_scale)
    Xdf['priority'] = Xdf['priority'].map(priority_scale)

    Xdf['number'] = Xdf['number'].apply(lambda x: x.strip('INC')).astype('i
    Xdf['opened_by'] = Xdf['opened_by'].apply(lambda x: x.strip('Openedby '
    Xdf['assigned_to'] = Xdf['assigned_to'].apply(lambda x: x.strip('Resolv

    cols_oe = ['contact_type', 'closed_code', 'incident_state', 'category',
               'caller_id', 'sys_updated_by', 'location', 'subcategory',
               'u_symptom', 'resolved_by']

    # ordinal encoder  for high cardinality columns
    oe = OrdinalEncoder()
    X_oe = pd.DataFrame(oe.fit_transform(Xdf[cols_oe]),
                                columns=cols_oe,
                                index=Xdf.index)
    Xdf = Xdf.drop(cols_oe, axis=1) # Drop original columns
    Xdf = pd.concat([Xdf, X_oe], axis=1) # Concat encoded cols with the ori

    return Xdf
```

In [34]:
```
incident_df1 = categorical_encoder(incident_df1)
```

In [35]:
```
incident_df1.head()
```

Out[35]:

	number	active	reassignment_count	reopen_count	sys_mod_count	made_sla	opened_by	ope
0	45	1	0	0	0	1	8	29/
1	45	1	0	0	2	1	8	29/
2	45	1	0	0	3	1	8	29/
3	45	0	0	0	4	1	8	29/
4	47	1	0	0	0	1	397	29/

3.2.7 datetime Features

In [36]:
```
# Function to convert date time columns recorded as strings to datetime (Fe
def date_time_converter(Xdf, datetime_cols):
    #datetime_cols = ['opened_at', 'sys_updated_at', 'resolved_at', 'closed
    # # Note: 'sys_created_at' was deleted due to presence of null vals.
    for col in datetime_cols:
        Xdf[col] = pd.to_datetime(Xdf[col])

    return Xdf
```

In [37]:
```
datetime_cols = ['opened_at', 'sys_updated_at', 'resolved_at', 'closed_at']
incident_df1 =  date_time_converter(incident_df1, datetime_cols)
```

In [38]:
```
incident_df1[datetime_cols].head()
```

Out[38]:

	opened_at	sys_updated_at	resolved_at	closed_at
0	2016-02-29 01:16:00	2016-02-29 01:23:00	2016-02-29 11:29:00	2016-05-03 12:00:00
1	2016-02-29 01:16:00	2016-02-29 08:53:00	2016-02-29 11:29:00	2016-05-03 12:00:00
2	2016-02-29 01:16:00	2016-02-29 11:29:00	2016-02-29 11:29:00	2016-05-03 12:00:00
3	2016-02-29 01:16:00	2016-05-03 12:00:00	2016-02-29 11:29:00	2016-05-03 12:00:00
4	2016-02-29 04:40:00	2016-02-29 04:57:00	2016-01-03 09:52:00	2016-06-03 10:00:00

3.3 Feature engineering

3.3.1 Feature extraction from datetime columns - Function

In [39]:
```
def extract_dt(Xdf, col_name):
    df = pd.DataFrame()
    df[col_name+'_year'] = Xdf[col_name].dt.year
    df[col_name+'_month'] = Xdf[col_name].dt.month

    df[col_name+'_week'] = Xdf[col_name].dt.isocalendar().week
    df[col_name+'_week'] = df[col_name+'_week'].apply(lambda x:1 if x==53 e

    df[col_name+'_day'] = Xdf[col_name].dt.day
    df[col_name+'_hour'] = Xdf[col_name].dt.hour
    df[col_name+'_minute'] = Xdf[col_name].dt.minute

    return df
```

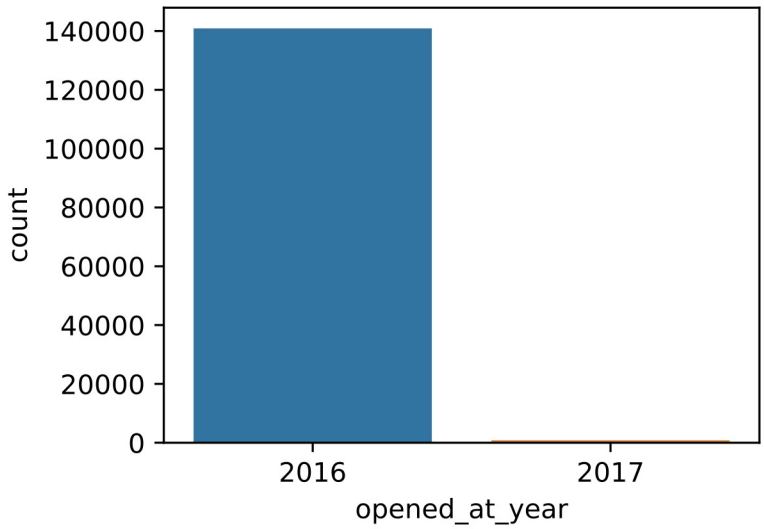
In [40]:
```
# Extract datetime features.
def feature_extract_dt(Xdf):
    datetime_cols = ['opened_at','resolved_at', 'sys_updated_at', 'closed_a
    new_features = []
    for col in datetime_cols:
        if Xdf[col].isna().sum() == 0:
            new_features.append(extract_dt(Xdf, col_name=col))
        else:
            #df_delta = Xdf['resolved_at'] - Xdf['opened_at']
            #delta = (Xdf['resolved_at'] - Xdf['opened_at']).mean()
            delta = datetime.timedelta(5,0,40,40.813301484)
            Xdf_shift = Xdf['opened_at'].apply(lambda x: x + delta)
            Xdf['resolved_at'] = Xdf['resolved_at'].fillna(value=Xdf_shift)
            new_features.append(extract_dt(Xdf, col_name=col))

    return new_features
```

In [41]:
```
# Feature extraction from datetime columns.
extracted_features = pd.concat(feature_extract_dt(incident_df1), axis=1)
incident_df1 = incident_df1.drop(datetime_cols, axis=1)
incident_df1 = pd.concat([incident_df1, extracted_features], axis=1)
```

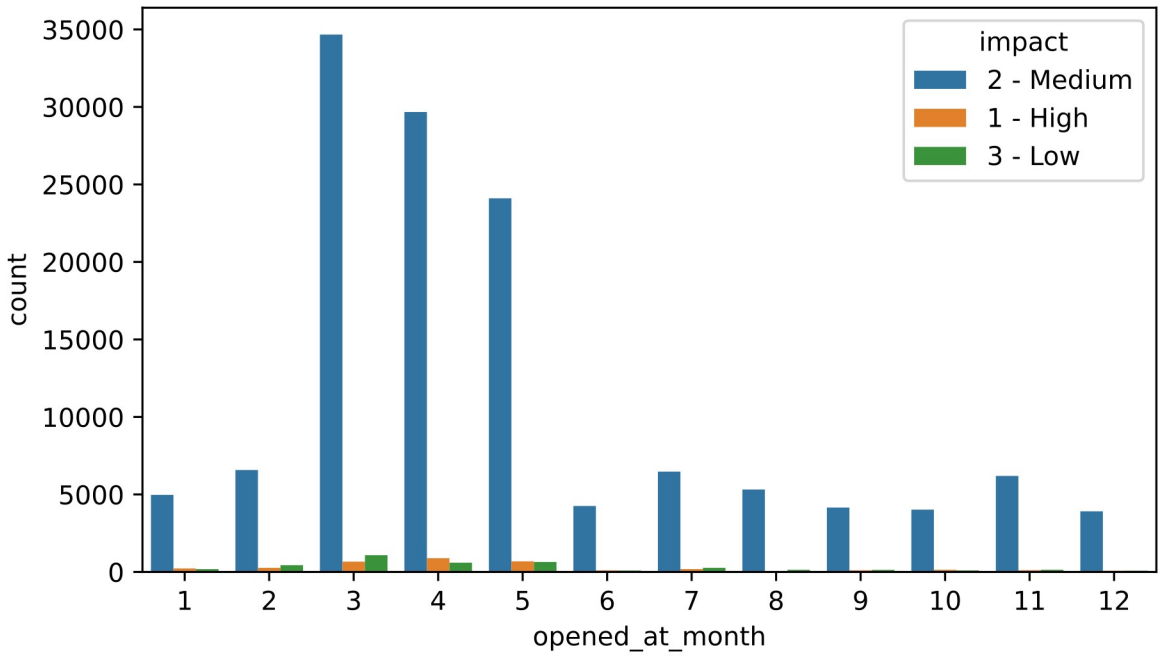
3.3.2 Features from 'opened at'

'opened_at_year', 'opened_at_month', 'opened_at_week', 'opened_at_day', 'opened_at_hour', 'opened_at_minute'

In [42]:
```
fig, ax = plt.subplots(figsize=(4,3))
sns.countplot(x=incident_df1['opened_at_year'], ax=ax)
plt.show()
```

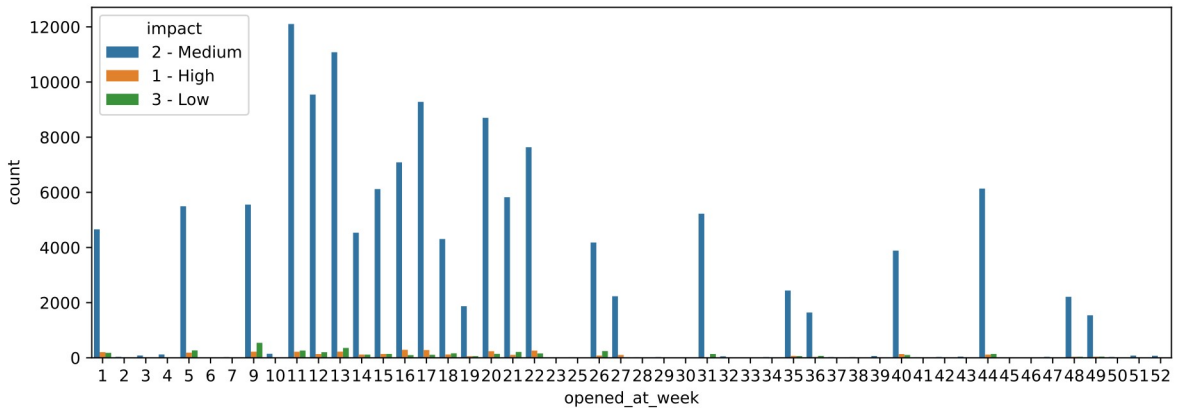


In [43]:
```
fig, ax = plt.subplots()
sns.countplot(x=incident_df1['opened_at_month'], ax=ax, hue=incident_df1['i
plt.show()
```



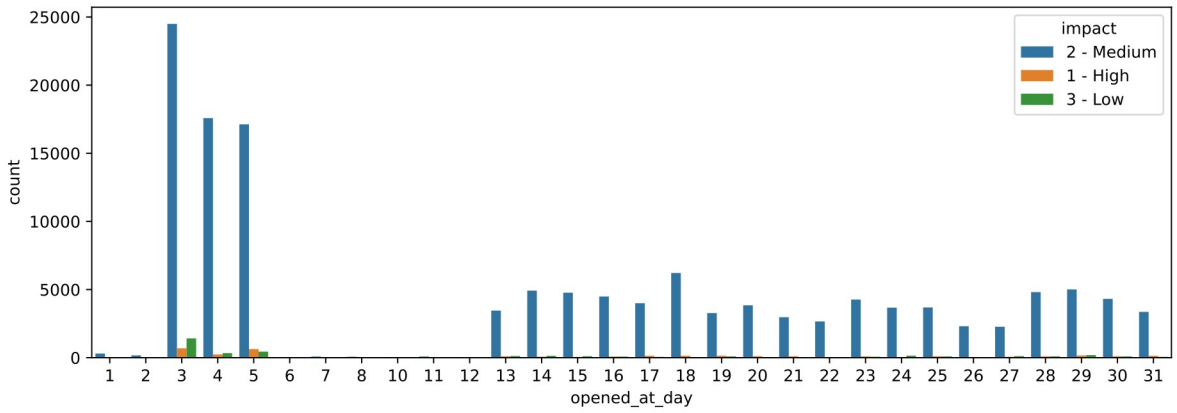
In [44]:

```
fig, ax = plt.subplots(figsize=(12,4))
sns.countplot(x=incident_df1['opened_at_week'], ax=ax, hue=incident_df1['im
plt.show()
```

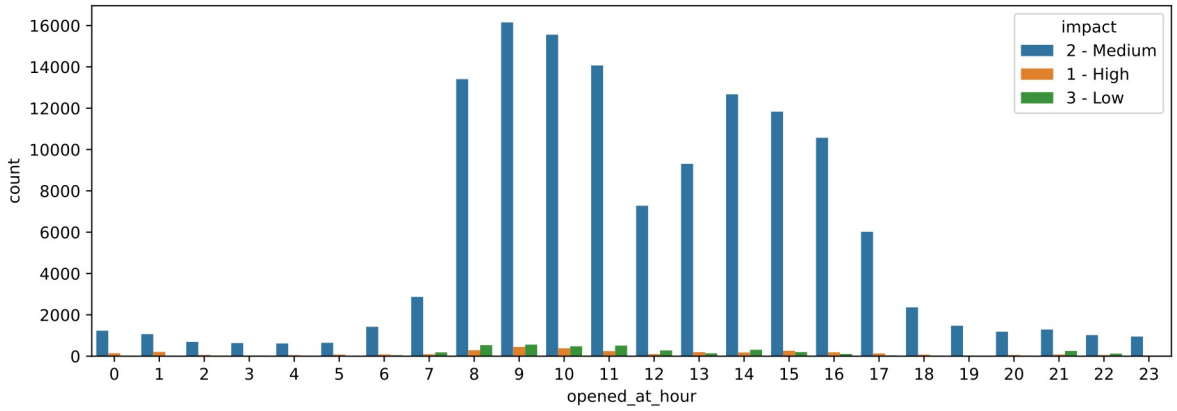


In [45]:

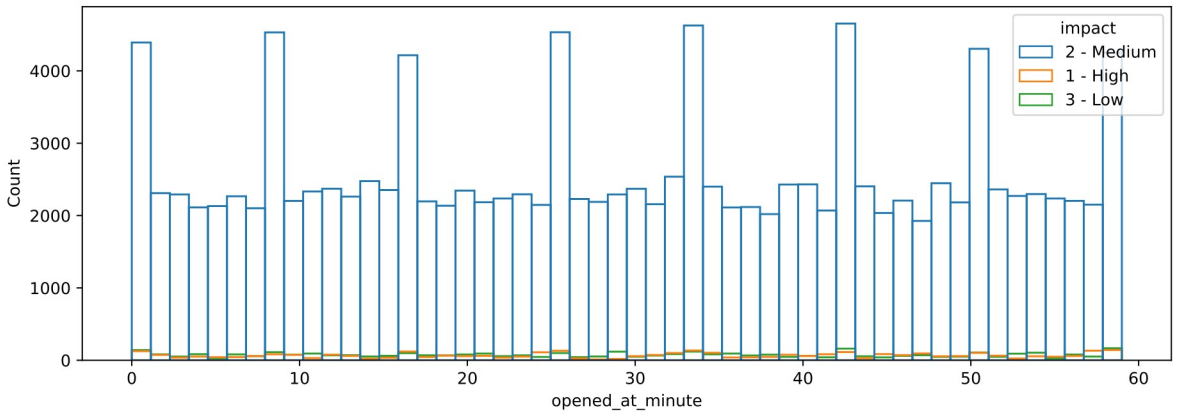
```
fig, ax = plt.subplots(figsize=(12,4))
sns.countplot(x=incident_df1['opened_at_day'], ax=ax, hue=incident_df1['imp
plt.show()
```



In [46]:
```
fig, ax = plt.subplots(figsize=(12,4))
sns.countplot(x=incident_df1['opened_at_hour'], ax=ax, hue=incident_df1['im
plt.show()
```



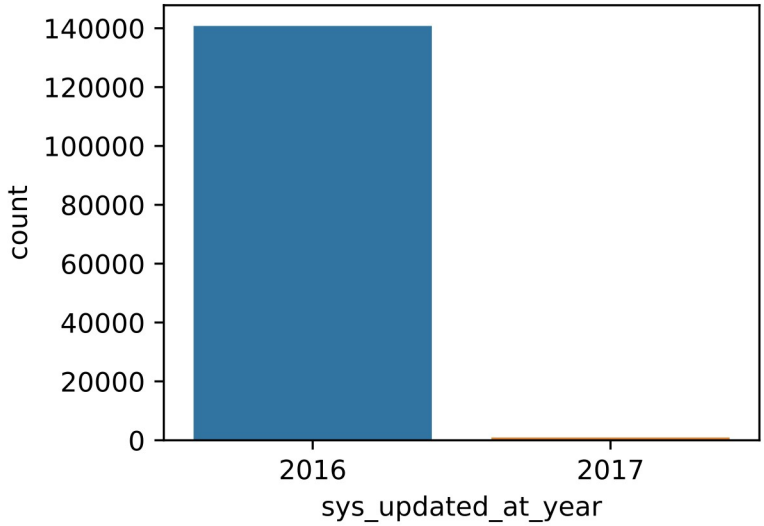
In [47]:
```
fig, ax = plt.subplots(figsize=(12,4))
sns.histplot(x=incident_df1['opened_at_minute'], ax=ax,hue=incident_df1['im
plt.show()
```



3.3.3 Features from 'sys_updated_at'

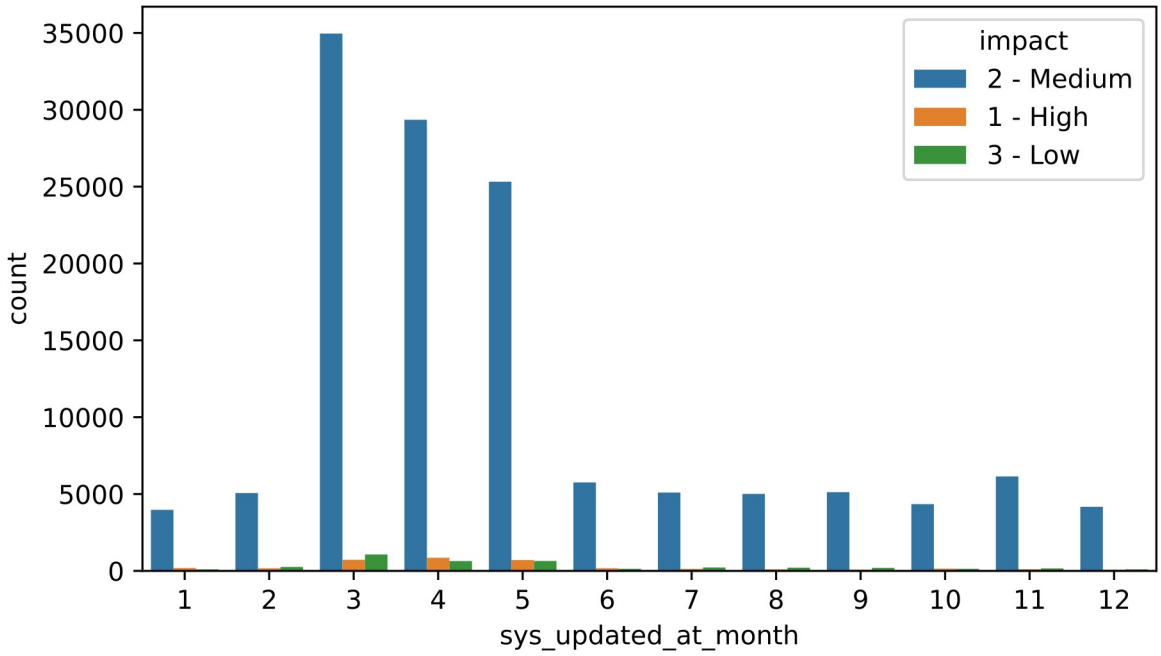
'sys_updated_at_year', 'sys_updated_at_month', 'sys_updated_at_week', 'sys_updated_at_day', 'sys_updated_at_hour', 'sys_updated_at_minute'

In [48]:
```
fig, ax = plt.subplots(figsize=(4,3))
sns.countplot(x=incident_df1['sys_updated_at_year'], ax=ax)
plt.show()
```



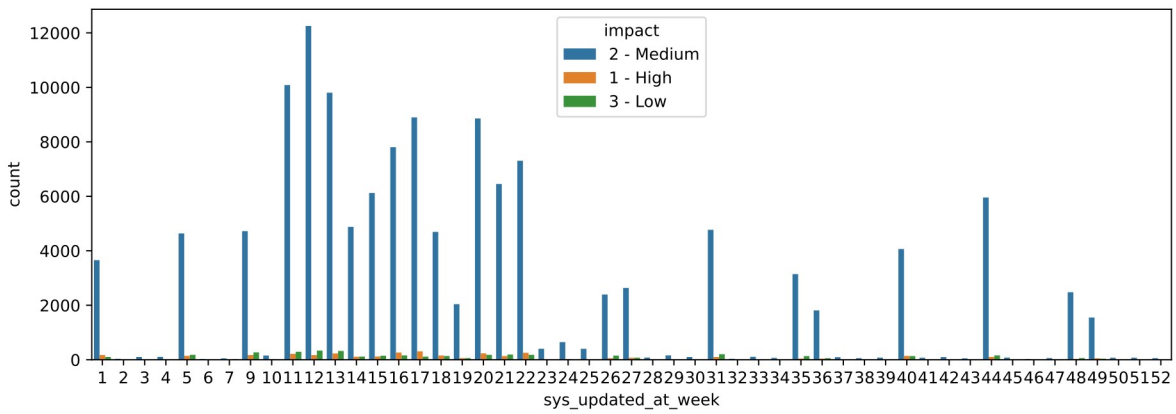
In [49]:

```
fig, ax = plt.subplots()
sns.countplot(x=incident_df1['sys_updated_at_month'], ax=ax, hue=incident_c
plt.show()
```

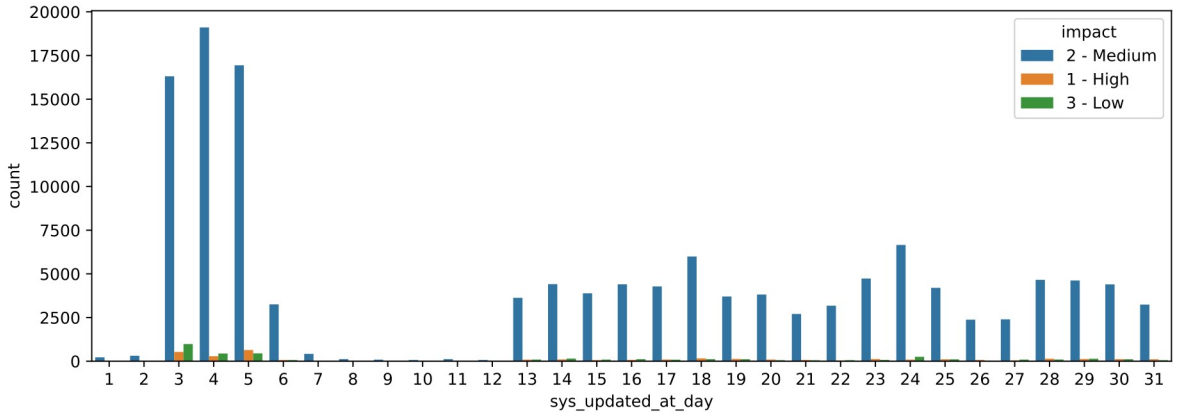


In [50]:

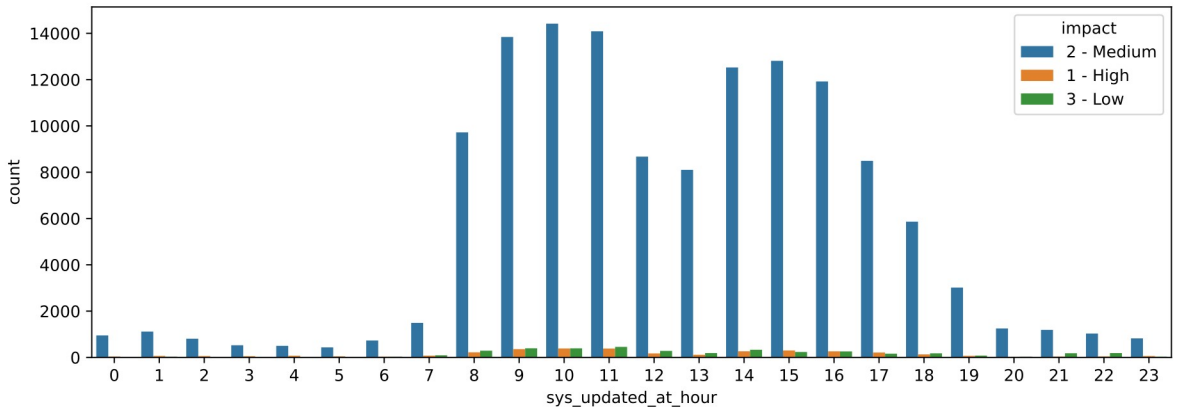
```
fig, ax = plt.subplots(figsize=(12,4))
sns.countplot(x=incident_df1['sys_updated_at_week'], ax=ax, hue=incident_df
plt.show()
```



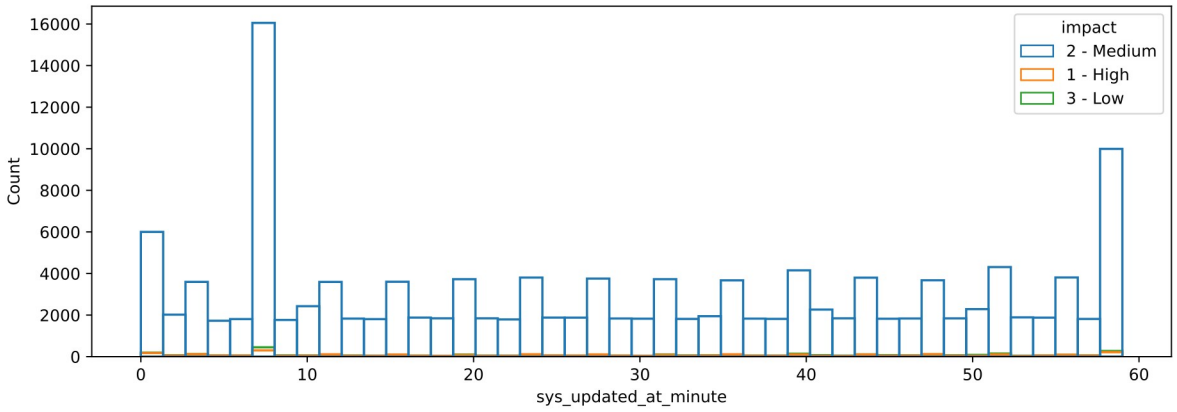
In [51]:
```
fig, ax = plt.subplots(figsize=(12,4))
sns.countplot(x=incident_df1['sys_updated_at_day'], ax=ax, hue=incident_df1
plt.show()
```



In [52]:
```
fig, ax = plt.subplots(figsize=(12,4))
sns.countplot(x=incident_df1['sys_updated_at_hour'], ax=ax, hue=incident_df
plt.show()
```



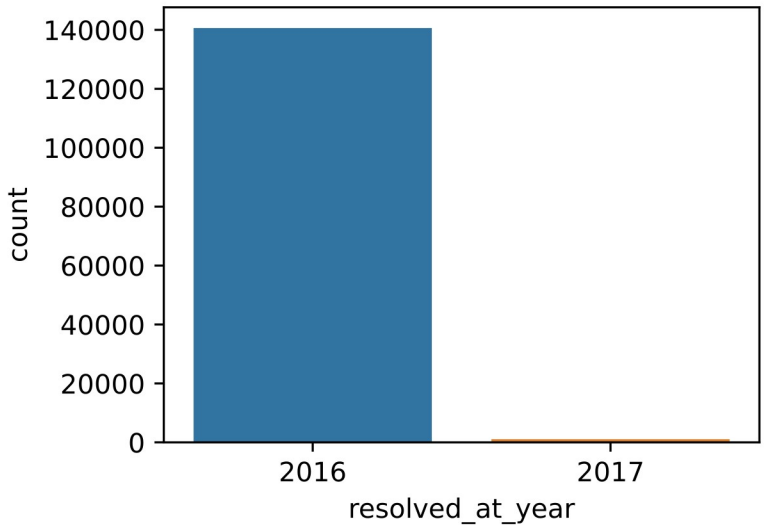
In [53]:
```
fig, ax = plt.subplots(figsize=(12,4))
sns.histplot(x=incident_df1['sys_updated_at_minute'], ax=ax,hue=incident_df
plt.show()
```



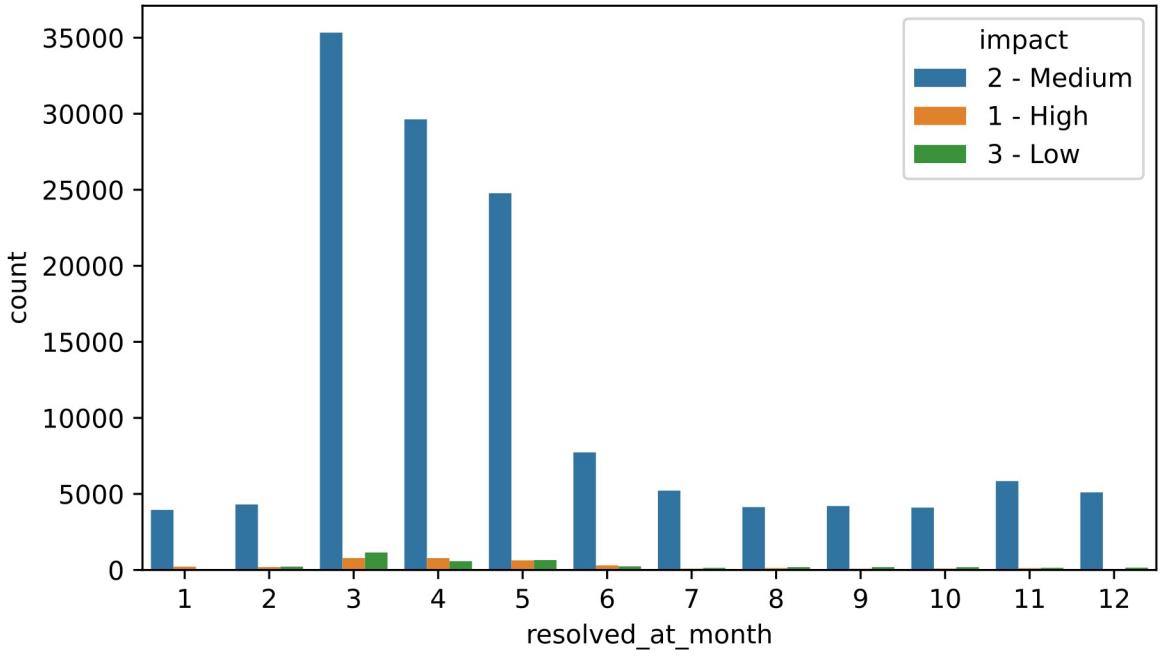
3.3.4 Features from 'resolved_at'

'resolved_at_year', 'resolved_at_month', 'resolved_at_week', 'resolved_at_day', 'resolved_at_hour', 'resolved_at_minute'

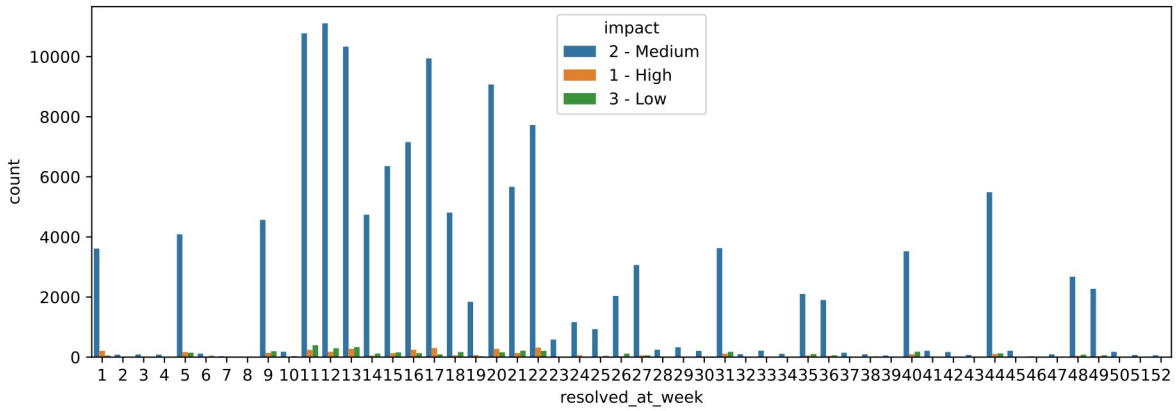
In [54]:
```
fig, ax = plt.subplots(figsize=(4,3))
sns.countplot(x=incident_df1['resolved_at_year'], ax=ax)
plt.show()
```



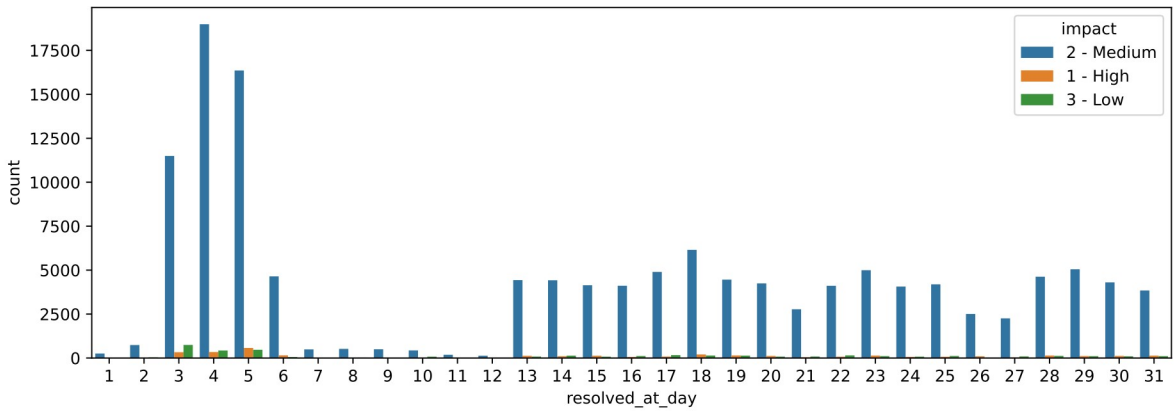
In [55]:
```
fig, ax = plt.subplots()
sns.countplot(x=incident_df1['resolved_at_month'], ax=ax, hue=incident_df1[
plt.show()
```



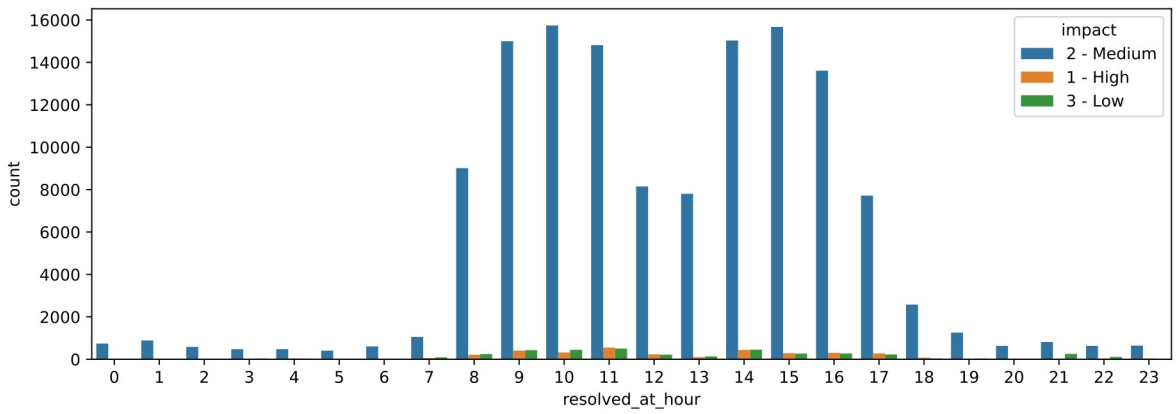
In [56]:
```
fig, ax = plt.subplots(figsize=(12,4))
sns.countplot(x=incident_df1['resolved_at_week'], ax=ax, hue=incident_df1['
plt.show()
```



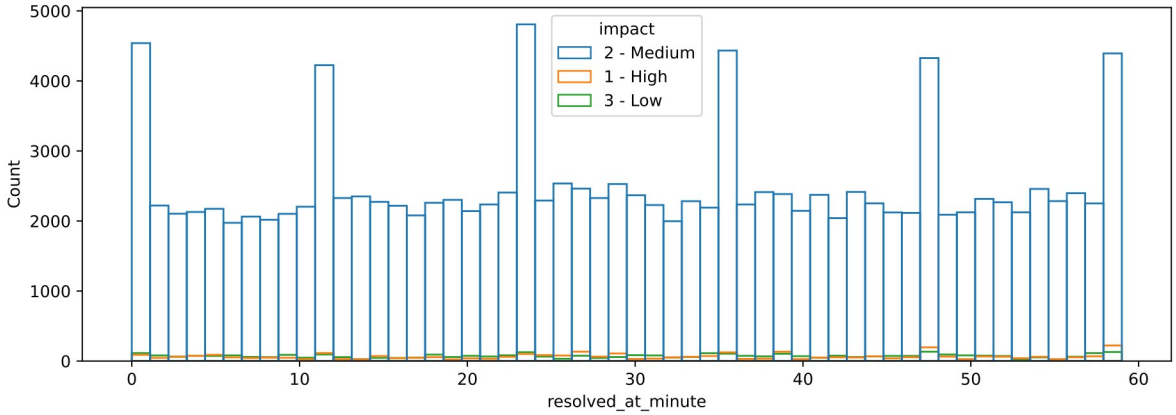
In [57]:
```
fig, ax = plt.subplots(figsize=(12,4))
sns.countplot(x=incident_df1['resolved_at_day'], ax=ax, hue=incident_df1['i
plt.show()
```



In [58]:
```
fig, ax = plt.subplots(figsize=(12,4))
sns.countplot(x=incident_df1['resolved_at_hour'], ax=ax, hue=incident_df1['
plt.show()
```



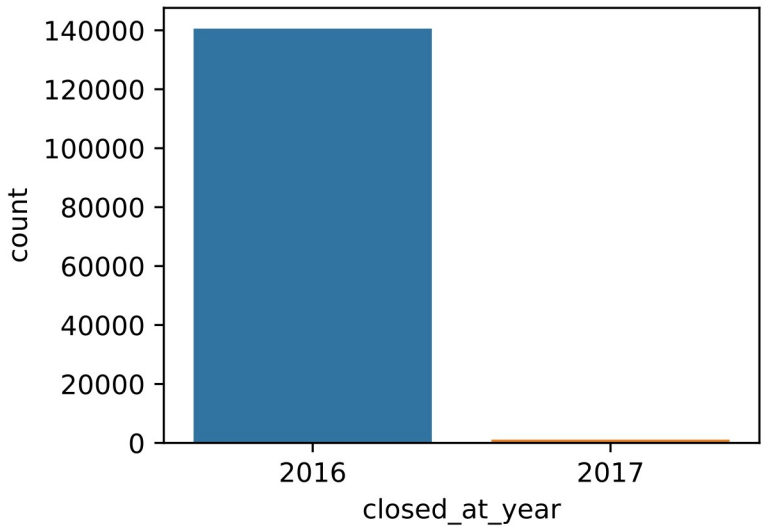
In [59]:
```
fig, ax = plt.subplots(figsize=(12,4))
sns.histplot(x=incident_df1['resolved_at_minute'], ax=ax, hue=incident_df1[
plt.show()
```



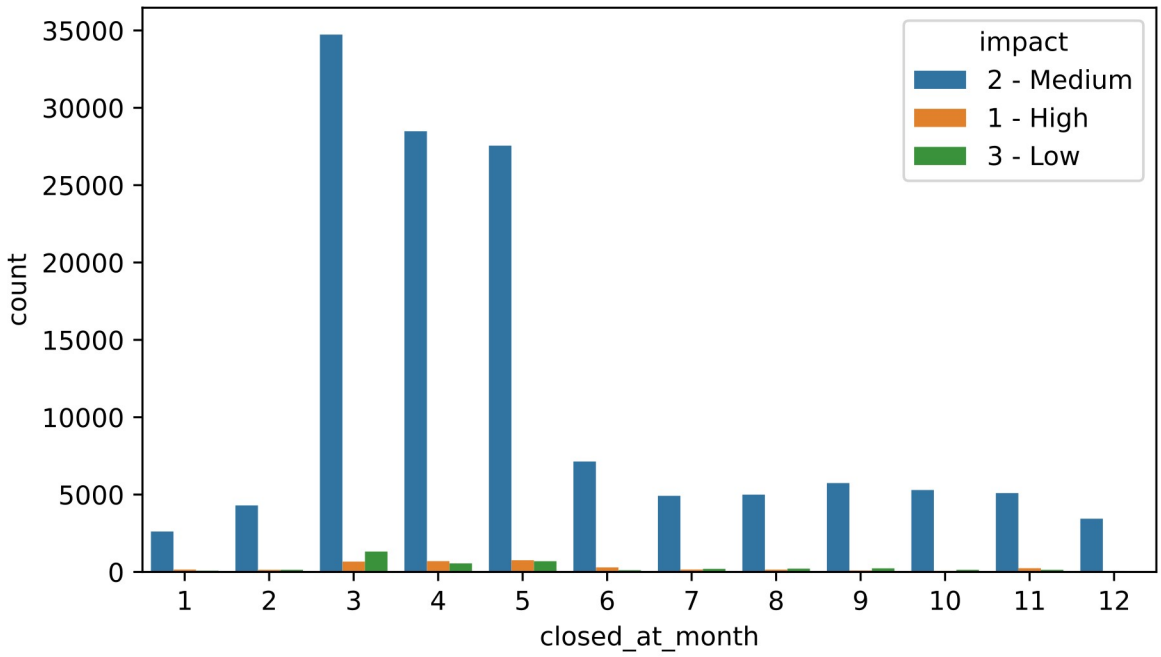
3.3.5 Features from 'closed_at'

'closed_at_year', 'closed_at_month', 'closed_at_week', 'closed_at_day', 'closed_at_hour', 'closed_at_minute'

In [60]:
```
fig, ax = plt.subplots(figsize=(4,3))
sns.countplot(x=incident_df1['closed_at_year'], ax=ax)
plt.show()
```

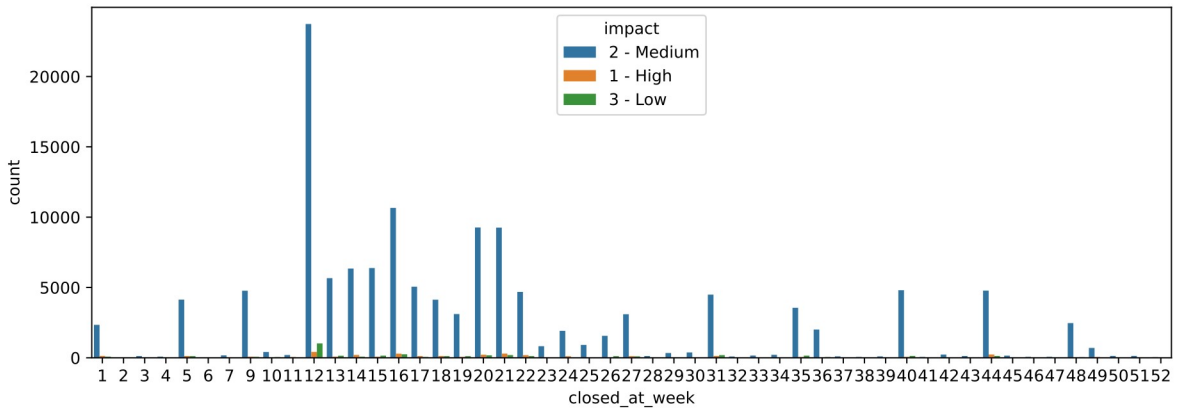


In [61]:
```
fig, ax = plt.subplots()
sns.countplot(x=incident_df1['closed_at_month'], ax=ax, hue=incident_df1['i
plt.show()
```



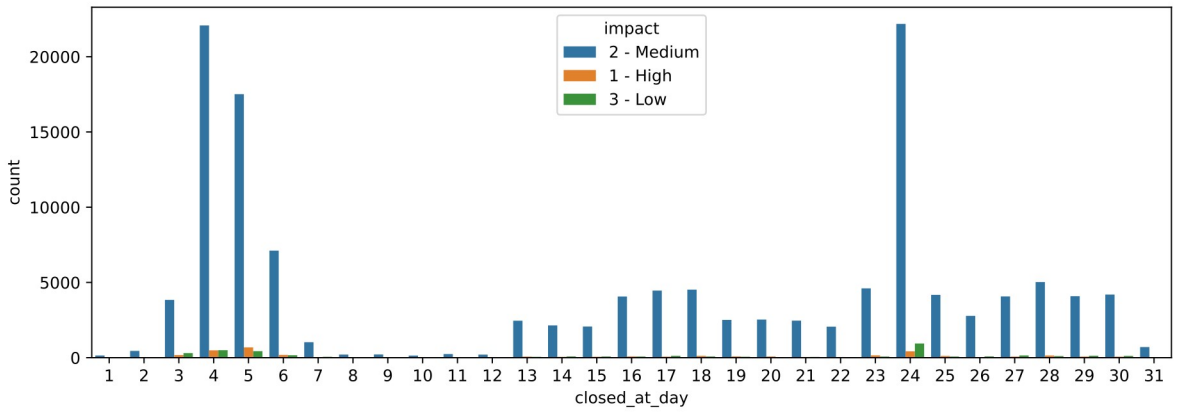
In [62]:

```
fig, ax = plt.subplots(figsize=(12,4))
sns.countplot(x=incident_df1['closed_at_week'], ax=ax, hue=incident_df1['im
plt.show()
```

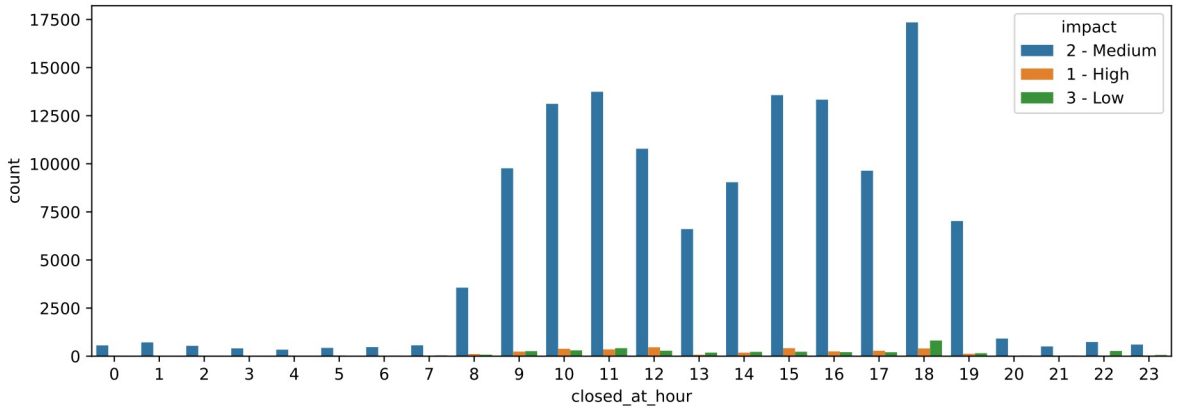


In [63]:

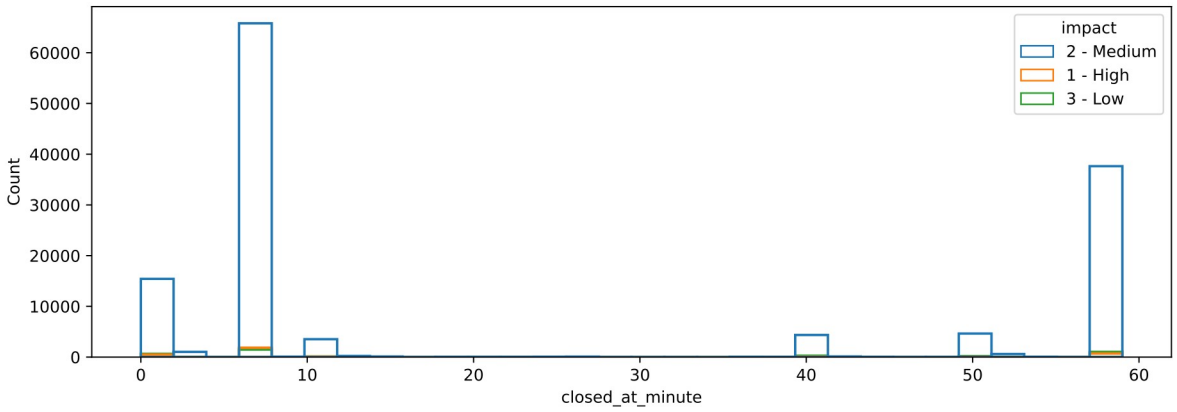
```
fig, ax = plt.subplots(figsize=(12,4))
sns.countplot(x=incident_df1['closed_at_day'], ax=ax, hue=incident_df1['imp
plt.show()
```



In [64]:
```
fig, ax = plt.subplots(figsize=(12,4))
sns.countplot(x=incident_df1['closed_at_hour'], ax=ax, hue=incident_df1['im
plt.show()
```



In [65]:
```
fig, ax = plt.subplots(figsize=(12,4))
sns.histplot(x=incident_df1['closed_at_minute'], ax=ax,hue=incident_df1['im
plt.show()
```



3.4 Encoding target labels

In [66]:
```
# Encoding the labels of the target column.
def out_label_encoder(ydf):
    impact_scale = {'1 - High':1, '2 - Medium':2, '3 - Low':3}
    ydf = ydf.map(impact_scale)
    return ydf
```

In [67]:
```
incident_df1['impact'] = out_label_encoder(incident_df1['impact'])
```

In [68]:
```
incident_df1.head()
```

Out[68]:

	number	active	reassignment_count	reopen_count	sys_mod_count	made_sla	opened_by	imp
0	45	1	0	0	0	1	8	
1	45	1	0	0	2	1	8	
2	45	1	0	0	3	1	8	
3	45	0	0	0	4	1	8	

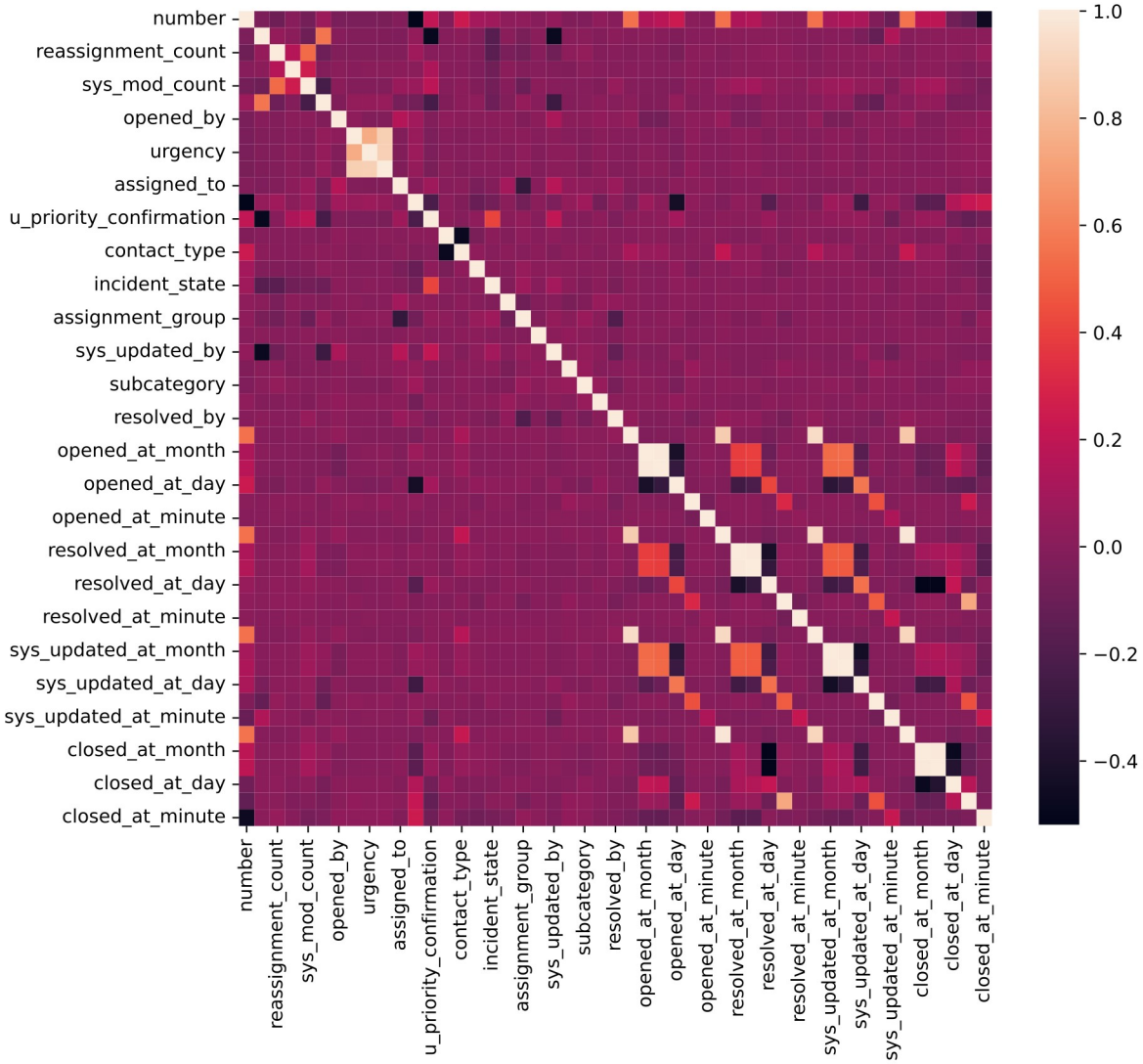
number	active	reassignment_count	reopen_count	sys_mod_count	made_sla	opened_by	imp

3.5 Correlation plots

3.5.1 Correlation between all features and target

In [69]:
```
incident_corr1 = incident_df1.corr()
```

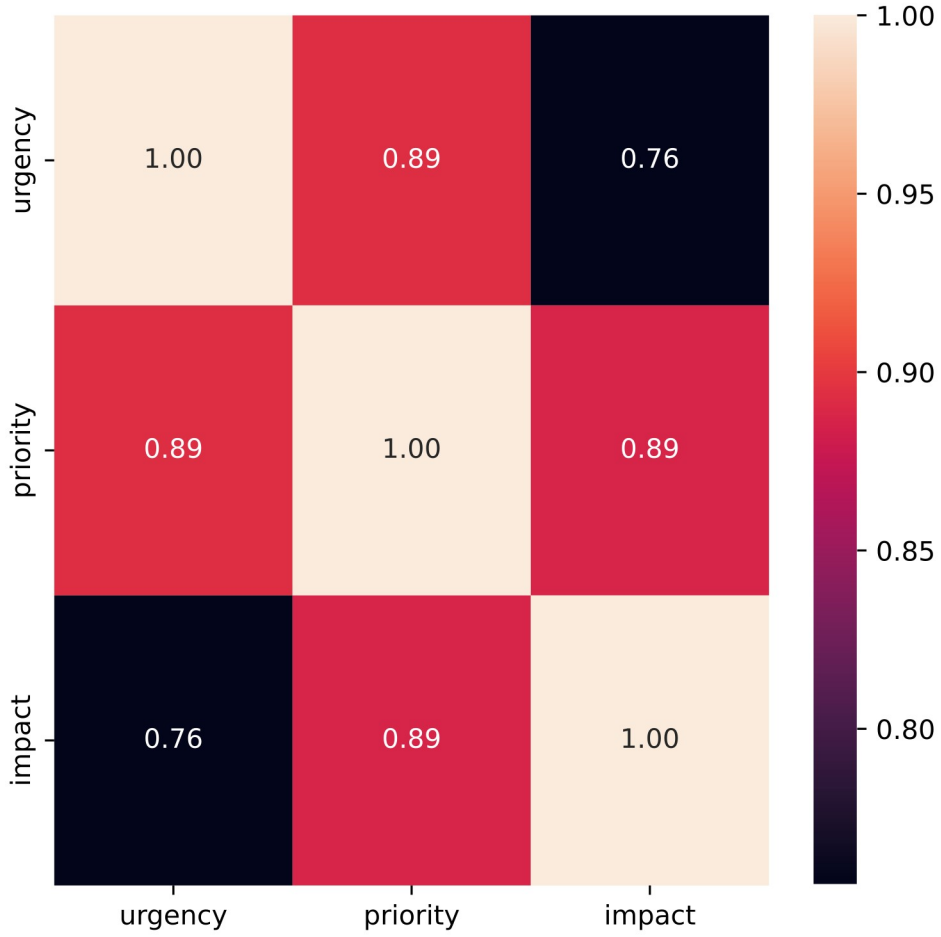
In [70]:
```
fig, ax = plt.subplots(figsize=(9,8))
sns.heatmap(incident_corr1, ax=ax, annot=False)
#fig.savefig('heatmap_incident.jpg', bbox_inches='tight', dpi=150) # TO sav
plt.show()
```



3.5.2 Correlation between 'urgency', 'priority' and 'impact'

In [71]:
```
incident_corr2 = incident_df1[['urgency', 'priority', 'impact']].corr()
```

In [72]:
```
fig, ax = plt.subplots(figsize=(6,6))
sns.heatmap(incident_corr2, ax=ax, annot=True, fmt='.2f')
#fig.savefig('heatmap_incident_lowcrd.jpg', bbox_inches='tight', dpi=150)
plt.show()
```



3.6 Expectation from the classifier for class predictions.

In [73]:
```
target_class_prob = incident_df1['impact'].value_counts().to_dict()
```

In [74]:
```
target_class_sumry = pd.concat([pd.Series(target_class_prob.keys()),
                                pd.Series(target_class_prob.values())],
                               axis=1)
target_class_sumry.columns = ['class', 'observations']
```

In [75]:
```
tot_num_obs = target_class_sumry['observations'].sum()
```

In [76]:
```
target_class_sumry['p_class'] = target_class_sumry['observations'].apply(la
```

In [77]:
```
target_class_sumry # Expectation
```

Out[77]:

	class	observations	p_class
0	2	134335	0.948
1	3	3886	0.027
2	1	3491	0.025

4. Data preparation

4.1 Splitting data into features and target.

In [78]:
```
target = 'impact'
features = [col for col in incident_df.columns if col != 'impact']
```

In [79]:
```
# data split
X = incident_df.loc[:, features]
y = incident_df.loc[:, target]
```

In [80]:
```
X.head()
```

Out[80]:

	number	incident_state	active	reassignment_count	reopen_count	sys_mod_count	made_sl
0	INC0000045	New	True	0	0	0	Tru
1	INC0000045	Resolved	True	0	0	2	Tru
2	INC0000045	Resolved	True	0	0	3	Tru
3	INC0000045	Closed	False	0	0	4	Tru
4	INC0000047	New	True	0	0	0	Tru

In [81]:
```
y.head()
```

Out[81]:
```
0    2 - Medium
1    2 - Medium
2    2 - Medium
3    2 - Medium
4    2 - Medium
Name: impact, dtype: object
```

4.2 Data preprocessing

In [82]:
```
# Final  data preprocessing function that includes the above preprocessing
def data_preprocessor(Xdf):
    Xdf =  null_value_remove(Xdf)
    Xdf =  null_value_impute(Xdf)
    Xdf =  boolean_encoder(Xdf)

    datetime_cols = ['opened_at', 'sys_updated_at', 'resolved_at', 'closed_
    Xdf =  date_time_converter(Xdf, datetime_cols)
    extracted_features = pd.concat(feature_extract_dt(Xdf), axis=1)
    Xdf = Xdf.drop(datetime_cols, axis=1)
    Xdf = pd.concat([Xdf, extracted_features], axis=1)

    Xdf = categorical_encoder(Xdf)

    return Xdf
```

In [83]:
```
%%time
# Preprocessing the features.
X_prep = data_preprocessor(X)
```

```
Wall time: 51.9 s
```

In [84]:
```
%%time
# Preprocessing the target.
y_prep = out_label_encoder(y)
```

```
Wall time: 22.4 ms
```

4.3 Train test split

In [85]:
```
# Splitting data into train and test set
X_train, X_test, y_train, y_test = train_test_split(X_prep, y_prep, test_si
```

In [86]:
```
X_train.shape, X_test.shape, y_train.shape, y_test.shape
```

Out[86]:
```
((113369, 48), (28343, 48), (113369,), (28343,))
```

In [87]:
```
X_train.head()
```

Out[87]:

	number	active	reassignment_count	reopen_count	sys_mod_count	made_sla	opened_by
132661	32475	1	0	0	1	1	55
123147	30070	0	1	0	10	0	8
81805	19472	1	0	0	0	1	24
24440	5443	1	1	0	2	1	30
18891	4120	1	3	0	7	1	59

In [88]:
```
X_test.head()
```

Out[88]:

	number	active	reassignment_count	reopen_count	sys_mod_count	made_sla	opened_by

	number	active	reassignment_count	reopen_count	sys_mod_count	made_sla	opened_by
121872	29727	1	0	0	1	1	24
5313	1287	1	0	0	2	1	468
65584	15348	1	0	0	0	1	17
137091	33660	0	0	0	4	1	13

In [89]:
```
test_df = pd.concat([X_test, y_test], axis=1)
```

In [90]:
```
test_df.head()
```

Out[90]:

	number	active	reassignment_count	reopen_count	sys_mod_count	made_sla	opened_by
121872	29727	1	0	0	1	1	24
5313	1287	1	0	0	2	1	468
65584	15348	1	0	0	0	1	17
137091	33660	0	0	0	4	1	13
96338	23275	1	0	0	0	1	24

In [91]:
```
test_df.to_csv("incident_impact_test.csv")
```

In [92]:
```
y_train[:5]
```

Out[92]:
```
132661    2
123147    2
81805     2
24440     2
18891     2
Name: impact, dtype: int64
```

In [93]:
```
y_test[y_test==1]
```

Out[93]:
```
95092     1
124909    1
92778     1
123526    1
458       1
         ..
31546     1
61995     1
65311     1
2999      1
129122    1
Name: impact, Length: 698, dtype: int64
```

4.4 Feature scaling

In [94]:
```
# Normalizing the data.
scaler = MinMaxScaler()
X_train = pd.DataFrame(scaler.fit_transform(X_train), columns=X_prep.columr
X_test = pd.DataFrame(scaler.transform(X_test), columns=X_prep.columns, inc
X_test.columns = X_prep.columns
```

In [95]:
```
X_train.head()
```

Out[95]:

	number	active	reassignment_count	reopen_count	sys_mod_count	made_sla	opened_b
132661	0.267974	1.0	0.000000	0.0	0.007752	1.0	0.09888
123147	0.248102	0.0	0.037037	0.0	0.077519	0.0	0.01119
81805	0.160529	1.0	0.000000	0.0	0.000000	1.0	0.04104
24440	0.044605	1.0	0.037037	0.0	0.015504	1.0	0.55783
18891	0.033672	1.0	0.111111	0.0	0.054264	1.0	0.10634

In [96]:
```
X_test.head()
```

Out[96]:

	number	active	reassignment_count	reopen_count	sys_mod_count	made_sla	opened_b
121872	0.245267	1.0	0.0	0.0	0.007752	1.0	0.04104
5313	0.010263	1.0	0.0	0.0	0.015504	1.0	0.86940
65584	0.126451	1.0	0.0	0.0	0.000000	1.0	0.02798
137091	0.277766	0.0	0.0	0.0	0.031008	1.0	0.24067
96338	0.191953	1.0	0.0	0.0	0.000000	1.0	0.04104

5. Model Building

5.1 Baseline model

Note: The preprocessed data is used and all features are considered for the baseline model.

In [97]:
```
# Function to initialize, train and give predictions
def clf_model(Xtrain, Xtest,  ytrain, ytest, classifier):
    """Fits the model specified in 'classifier' and returns
    predictions for both train and test data sets along with
    the fitted model.
    Inputs:
    ------
    Xtrain, Xtest,  ytrain, ytest, classifier

    outputs:
    ------
    ypred_train, ypred_test, clf
    """
    clf = classifier
    clf.fit(Xtrain, ytrain)

    ypred_train = clf.predict(Xtrain)
    ypred_test = clf.predict(Xtest)

    # Train data performance
    #display_results(ytrain, ypred_train, clf)
    #display_results(ytest, ypred_test, clf)
    return ypred_train, ypred_test, clf
```

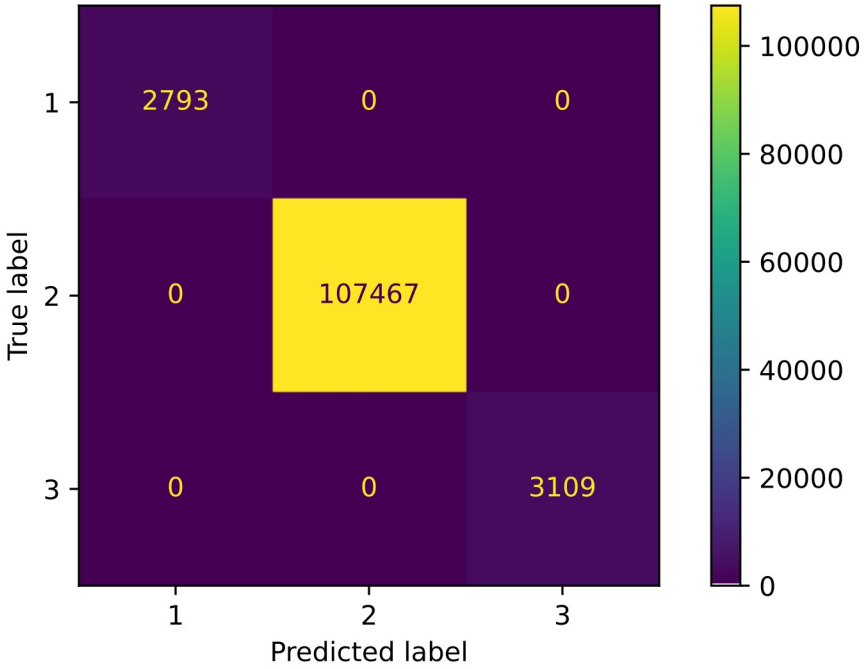
In [98]:
```
# Function to evaluate model
def display_results(y_test, y_pred, clf):
    """Displays model evaluation/performance report that includes
    accuracy_score, confusion_matrix, precision_score, and
    recall_score.
    input
    -----
    y_test, y_pred

    output
    ------
    Model evaluation/performance report"""
    print(classification_report(y_test, y_pred))
    #print("Confusion matrix:\n", confusion_matrix(y_test, y_pred))
    fig, ax = plt.subplots()
    ConfusionMatrixDisplay.from_predictions(y_test, y_pred, labels=clf.clas
```

In [99]:
```
# M1 decision tree all features considered.
y_pred_train, y_pred_test, clf =clf_model(Xtrain=X_train,
          Xtest=X_test,
          ytrain=y_train,
          ytest=y_test,
          classifier=DecisionTreeClassifier(random_state=42))
```

In [100…
```
display_results(y_train, y_pred_train, clf)
```

```
              precision    recall  f1-score   support

           1       1.00      1.00      1.00      2793
           2       1.00      1.00      1.00    107467
           3       1.00      1.00      1.00      3109

    accuracy                           1.00    113369
```

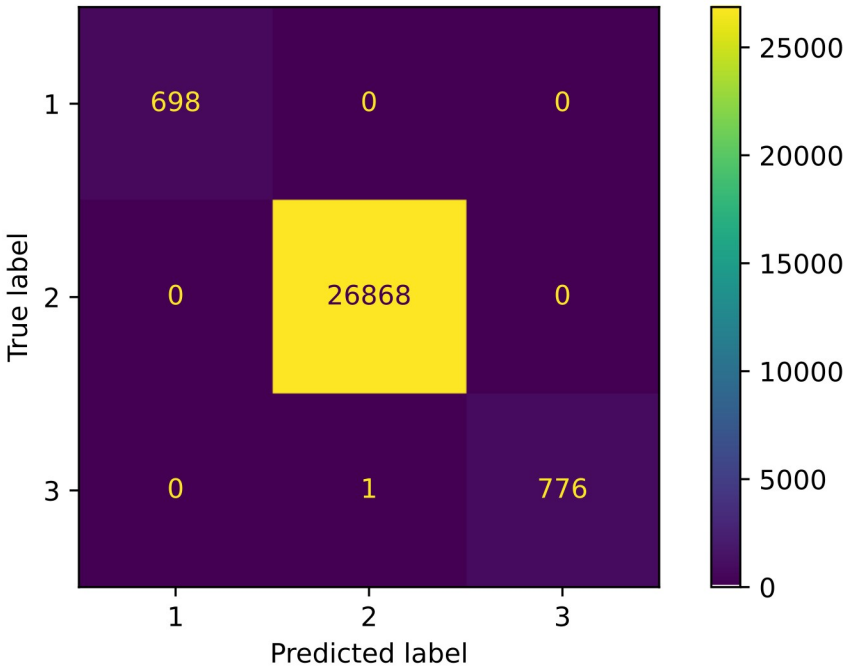
```
   macro avg       1.00      1.00      1.00    113369
weighted avg       1.00      1.00      1.00    113369
```



In [101…

```
display_results(y_test, y_pred_test, clf)
```

```
              precision    recall  f1-score   support

           1       1.00      1.00      1.00       698
           2       1.00      1.00      1.00     26868
           3       1.00      1.00      1.00       777

    accuracy                           1.00     28343
   macro avg       1.00      1.00      1.00     28343
weighted avg       1.00      1.00      1.00     28343
```



In [102…
```
def weighted_avg_scores(y_test, y_pred_test):
    scores = []
    prec_bs = round(precision_score(y_test, y_pred_test, average='weighted'
    rec_bs = round(recall_score(y_test, y_pred_test, average='weighted'),3)
    f1_bs = round(f1_score(y_test, y_pred_test, average='weighted'),3)
    acc_bs = round(accuracy_score(y_test, y_pred_test))
    scores = [prec_bs, rec_bs, f1_bs, acc_bs]
    return scores
```

In [103…
```
scores_bs = weighted_avg_scores(y_test, y_pred_test)
```

In [104…
```
scores_bs
```

Out[104… `[1.0, 1.0, 1.0, 1]`

In [105…
```
#metrics = ['precision', 'recall', 'f1_score', 'accuracy']
```

5.2 Feature selection

5.2.1 Chi2 Method

In [106…
```
# Feature Extraction with Univariate Statistical Chi-squared Test for class
from numpy import set_printoptions
from sklearn.feature_selection import SelectKBest
from sklearn.feature_selection import chi2

# feature extraction
test = SelectKBest(score_func=chi2, k=10)
fit = test.fit(X_train, y_train)
# summarize scores
#set_printoptions(precision=3)
#print(fit.scores_)
sel_features = fit.transform(X_train)
```

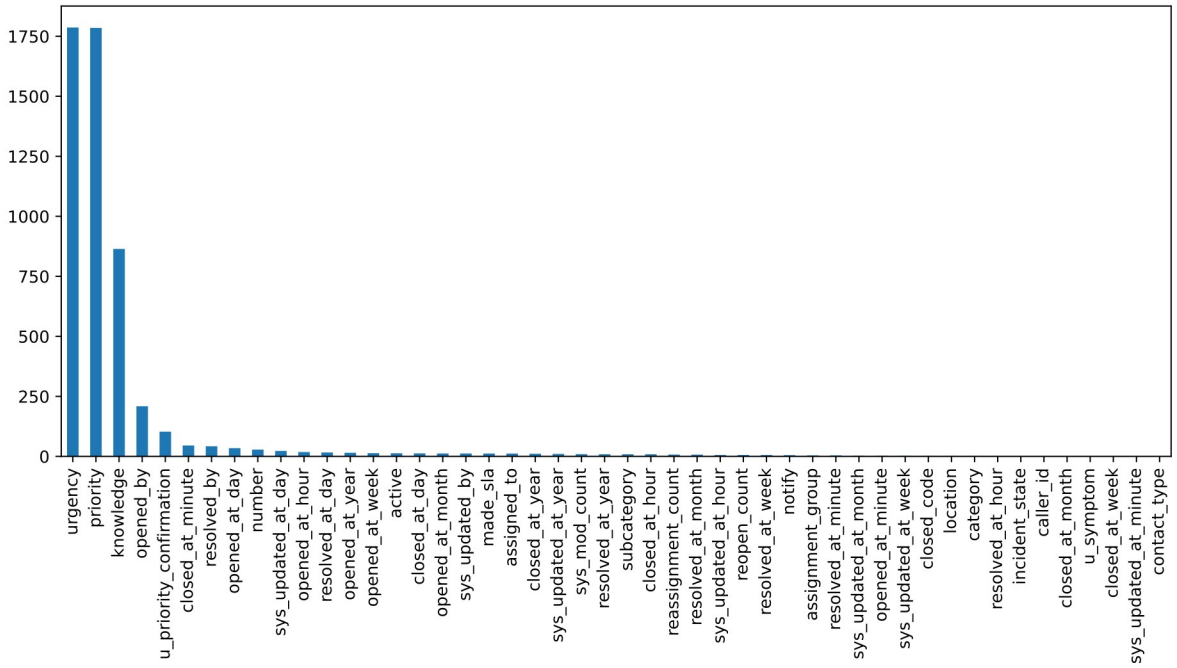
In [107…
```
# Summarixing the scores
scores = fit.scores_
scores = pd.Series(scores)
scores.index = X_prep.columns
scores = scores.sort_values(ascending=False)
```

In [108…
```
scores[:10]
```

Out[108…
```
urgency                   1786.577631
priority                  1784.770478
knowledge                  864.118637
opened_by                  208.896875
u_priority_confirmation    102.959102
closed_at_minute            45.474450
resolved_by                 42.162729
opened_at_day               34.478321
number                      28.300496
sys_updated_at_day          22.844291
dtype: float64
```

In [109…
```
plt.figure(figsize=(12,5))
scores.plot.bar()
```

Out[109…
```
<AxesSubplot:>
```



5.2.2 Mutual information classification

In [110…
```
from sklearn.feature_selection import mutual_info_classif
fs = SelectKBest(score_func=mutual_info_classif, k=10)
fs.fit(X_train, y_train)
X_train_fs = fs.transform(X_train)
#print(fs.scores_)
```

In [111…
```
X_prep.columns[fs.get_support()]
```

Out[111…
```
Index(['number', 'opened_by', 'urgency', 'priority', 'assigned_to',
       'caller_id', 'sys_updated_by', 'subcategory', 'u_symptom',
       'resolved_by'],
      dtype='object')
```

5.2.3 Feature importance using decision tree

In [112…
```
from sklearn.tree import  DecisionTreeClassifier
model = DecisionTreeClassifier(random_state=42)
fit=model.fit(X_train, y_train)
print(model.feature_importances_)
```

```
[5.60548007e-03 0.00000000e+00 9.28989907e-04 0.00000000e+00
 0.00000000e+00 0.00000000e+00 2.95389964e-03 1.04748525e-01
 8.73743025e-01 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 7.26229195e-04 1.25769877e-03 3.91834004e-04
 8.45962256e-04 0.00000000e+00 1.65953225e-04 0.00000000e+00
 0.00000000e+00 0.00000000e+00 0.00000000e+00 0.00000000e+00
 0.00000000e+00 2.44698592e-03 1.72798796e-04 2.33631635e-04
 0.00000000e+00 0.00000000e+00 0.00000000e+00 0.00000000e+00
```

```
1.13536292e-03 0.00000000e+00 1.18106001e-03 0.00000000e+00
0.00000000e+00 2.90064264e-03 4.57139671e-04 1.04781306e-04]
```

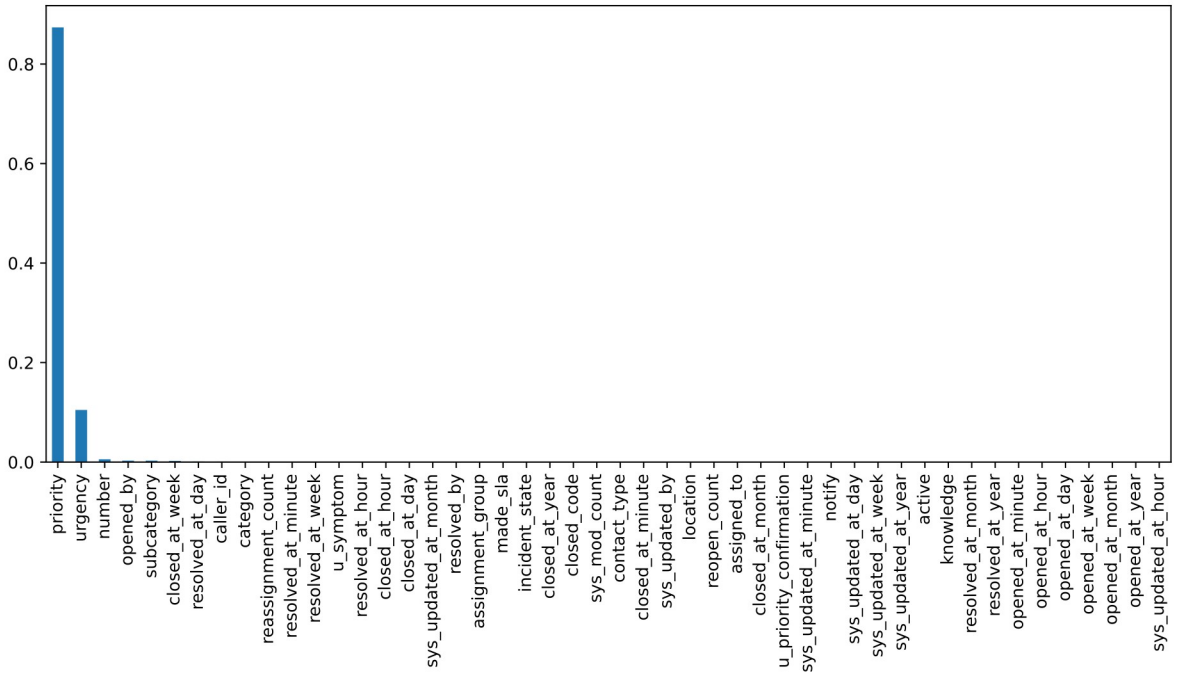
In [113…

```
features_imp=model.feature_importances_
features_imp=pd.Series(features_imp)
features_imp.index=X_prep.columns
features_imp=features_imp.sort_values(ascending=False)
```

In [114…

```
plt.figure(figsize=(12,5))
features_imp.plot.bar()
```

Out[114… <AxesSubplot:>



In [115…

```
print(features_imp[:10])
```

```
priority              0.873743
urgency               0.104749
number                0.005605
opened_by             0.002954
subcategory           0.002901
closed_at_week        0.002447
resolved_at_day       0.001258
caller_id             0.001181
category              0.001135
reassignment_count    0.000929
dtype: float64
```

5.2.4 Final set of features

In [116…

```
#Top 15 selected features from each model
chi2_features=scores[:10].to_dict()
chi2_features=set(chi2_features.keys())

mutualinfo_features=set(X_prep.columns[fs.get_support()])

rfe_features=features_imp[:13].to_dict()
rfe_features=set(rfe_features.keys())
```

In [117…
```
features_imp[:10].keys()
```

Out[117…
```
Index(['priority', 'urgency', 'number', 'opened_by', 'subcategory',
       'closed_at_week', 'resolved_at_day', 'caller_id', 'category',
       'reassignment_count'],
      dtype='object')
```

In [118…
```
#Use intersection fuction to get the common features from all the three set
chi2_features = chi2_features.intersection(mutualinfo_features)
output_set = chi2_features.intersection(rfe_features)
output_list = list(output_set)
print("Final list of features: ",output_list)
```

```
Final list of features:  ['urgency', 'number', 'opened_by', 'priority']
```

5.3 Training and validation with different models

In [119…
```
X_train.head()
```

Out[119…

	number	active	reassignment_count	reopen_count	sys_mod_count	made_sla	opened_b
132661	0.267974	1.0	0.000000	0.0	0.007752	1.0	0.09888
123147	0.248102	0.0	0.037037	0.0	0.077519	0.0	0.01119
81805	0.160529	1.0	0.000000	0.0	0.000000	1.0	0.04104
24440	0.044605	1.0	0.037037	0.0	0.015504	1.0	0.55783
18891	0.033672	1.0	0.111111	0.0	0.054264	1.0	0.10634

In [120…
```
y_train[:5]
```

Out[120…
```
132661    2
123147    2
81805     2
24440     2
18891     2
Name: impact, dtype: int64
```

In [121…
```
sel_features = output_list # ['number', 'priority', 'urgency', 'opened_by']
```

In [122…
```
X_train1 = X_train[sel_features].copy()
X_test1 = X_test[sel_features].copy()
```

In [123…
```
X_test.head()
```

Out[123…

	number	active	reassignment_count	reopen_count	sys_mod_count	made_sla	opened_b
121872	0.245267	1.0	0.0	0.0	0.007752	1.0	0.04104
5313	0.010263	1.0	0.0	0.0	0.015504	1.0	0.86940
65584	0.126451	1.0	0.0	0.0	0.000000	1.0	0.02798
137091	0.277766	0.0	0.0	0.0	0.031008	1.0	0.24067

number	active	reassignment_count	reopen_count	sys_mod_count	made_sla	opened_b

5.3.1 Decision tree classifier

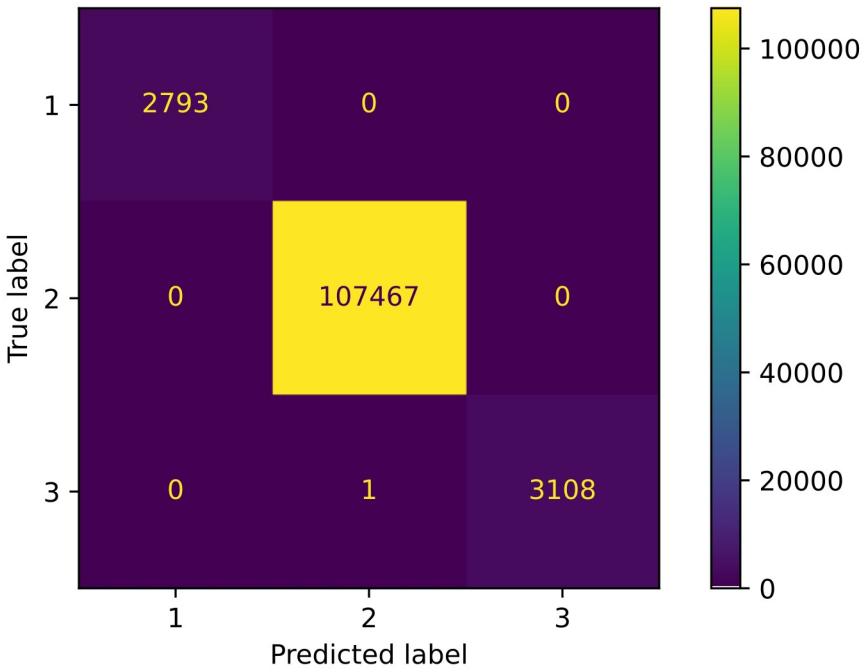
In [124...

```
y_pred_train_dt, y_pred_test_dt, clf_dt =clf_model(Xtrain=X_train1,
          Xtest=X_test1,
          ytrain=y_train,
          ytest=y_test,
          classifier=DecisionTreeClassifier(random_state=42))
```

In [125...

```
display_results(y_train, y_pred_train_dt, clf_dt)
```

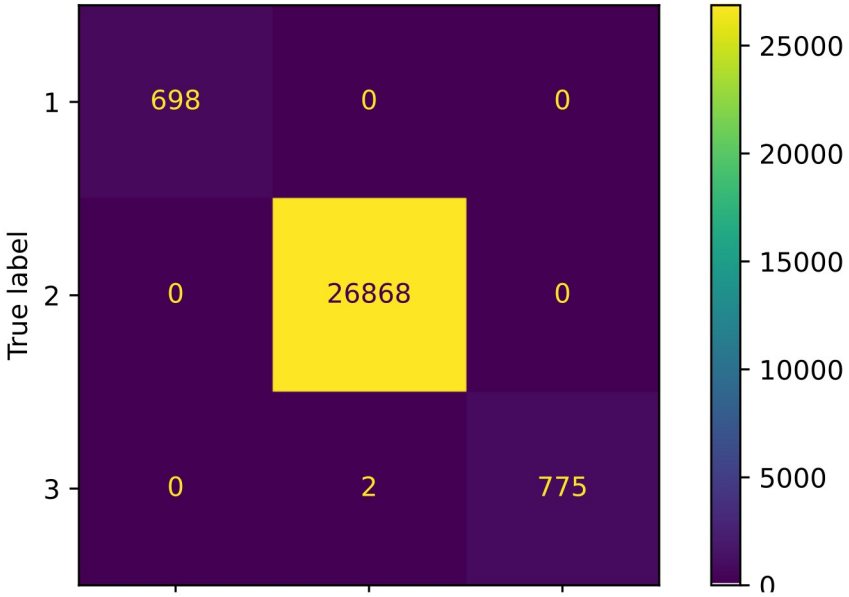
```
              precision    recall  f1-score   support

           1       1.00      1.00      1.00      2793
           2       1.00      1.00      1.00    107467
           3       1.00      1.00      1.00      3109

    accuracy                           1.00    113369
   macro avg       1.00      1.00      1.00    113369
weighted avg       1.00      1.00      1.00    113369
```



In [126...

```
display_results(y_test, y_pred_test_dt, clf_dt)
```

```
              precision    recall  f1-score   support

           1       1.00      1.00      1.00       698
           2       1.00      1.00      1.00     26868
           3       1.00      1.00      1.00       777

    accuracy                           1.00     28343
   macro avg       1.00      1.00      1.00     28343
weighted avg       1.00      1.00      1.00     28343
```



5.3.2 Random Forest classifier

In [127…

```
y_pred_train_rf, y_pred_test_rf, clf_rf =clf_model(Xtrain=X_train1,
          Xtest=X_test1,
          ytrain=y_train,
          ytest=y_test,
          classifier=RandomForestClassifier(random_state=42))
```

In [128…

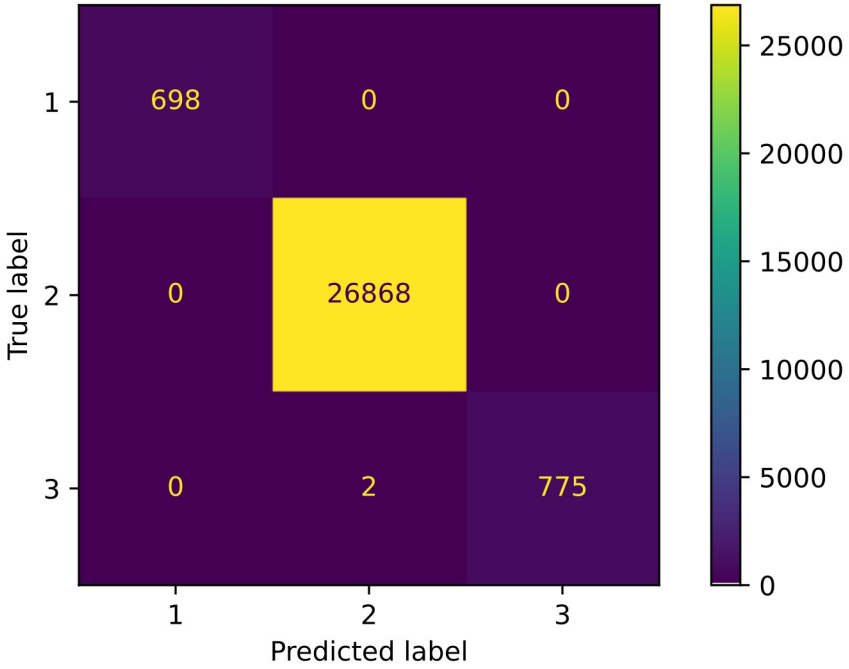
```
display_results(y_train, y_pred_train_rf, clf_rf)
```

```
              precision    recall  f1-score   support

           1       1.00      1.00      1.00      2793
           2       1.00      1.00      1.00    107467
           3       1.00      1.00      1.00      3109

    accuracy                           1.00    113369
   macro avg       1.00      1.00      1.00    113369
weighted avg       1.00      1.00      1.00    113369
```

In [129…

```
display_results(y_test, y_pred_test_rf, clf_rf)
```

```
              precision    recall  f1-score   support

           1       1.00      1.00      1.00       698
           2       1.00      1.00      1.00     26868
           3       1.00      1.00      1.00       777

    accuracy                           1.00     28343
   macro avg       1.00      1.00      1.00     28343
weighted avg       1.00      1.00      1.00     28343
```



5.3.3 XGBoost classifier

In [130…

```
y_pred_train_xb, y_pred_test_xb, clf_xb =clf_model(Xtrain=X_train1,
          Xtest=X_test1,
          ytrain=y_train,
          ytest=y_test,
          classifier=XGBClassifier(random_state=42))
```

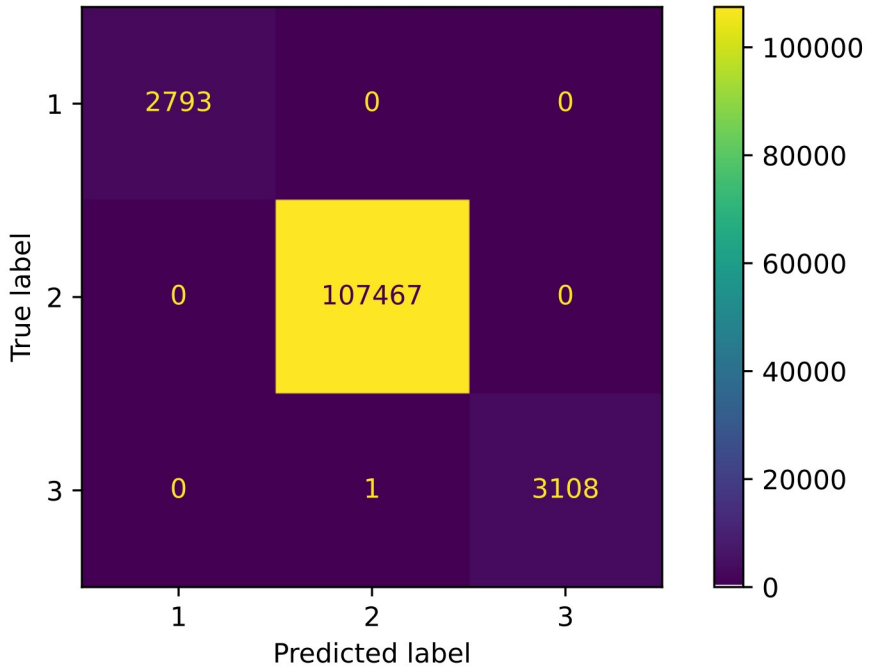
```
D:\anaconda3\lib\site-packages\xgboost\sklearn.py:1224: UserWarning: The us
e of label encoder in XGBClassifier is deprecated and will be removed in a
future release. To remove this warning, do the following: 1) Pass option us
e_label_encoder=False when constructing XGBClassifier object; and 2) Encode
your labels (y) as integers starting with 0, i.e. 0, 1, 2, ..., [num_class
- 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)
[16:16:04] WARNING: C:/Users/Administrator/workspace/xgboost-win64_release_
1.5.0/src/learner.cc:1115: Starting in XGBoost 1.3.0, the default evaluatio
n metric used with the objective 'multi:softprob' was changed from 'merror'
to 'mlogloss'. Explicitly set eval_metric if you'd like to restore the old
behavior.
```

In [131…

```
display_results(y_train, y_pred_train_xb, clf_xb)
```

```
              precision    recall  f1-score   support
```

1	1.00	1.00	1.00	2793
2	1.00	1.00	1.00	107467
3	1.00	1.00	1.00	3109
accuracy			1.00	113369
macro avg	1.00	1.00	1.00	113369
weighted avg	1.00	1.00	1.00	113369



In [132…

```
display_results(y_test, y_pred_test_xb, clf_xb)
```

	precision	recall	f1-score	support
1	1.00	1.00	1.00	698
2	1.00	1.00	1.00	26868
3	1.00	1.00	1.00	777
accuracy			1.00	28343
macro avg	1.00	1.00	1.00	28343
weighted avg	1.00	1.00	1.00	28343

5.3 SMOTE for balancing the data (Sampling)

In [133…

```
from imblearn.over_sampling import SMOTE
oversample=SMOTE()
X_balanced,y_balanced=oversample.fit_resample(X_train,y_train)
y_balanced=pd.Series(y_balanced)

print("Impact counts before balancing:\n ",y_train.value_counts())
print("Impact counts after balancing: \n",y_balanced.value_counts())
```

```
Impact counts before balancing:
  2    107467
3       3109
1       2793
Name: impact, dtype: int64
Impact counts after balancing:
 2    107467
3    107467
1    107467
Name: impact, dtype: int64
```

In [134…

```
X_balanced.shape
```

Out[134…

```
(322401, 48)
```

5.4 Training and validation with different models after oversampling

5.4.1 Decision tree

In [135…

```
X_train_os = X_balanced[sel_features].copy()
y_train_os = y_balanced.copy()
```

In [136…

```
y_pred_train_os_dt, y_pred_test_os_dt, clf_os_dt =clf_model(Xtrain=X_train_
          Xtest=X_test1,
          ytrain=y_train_os,
          ytest=y_test,
          classifier=DecisionTreeClassifier(random_state=42))
```

In [137…

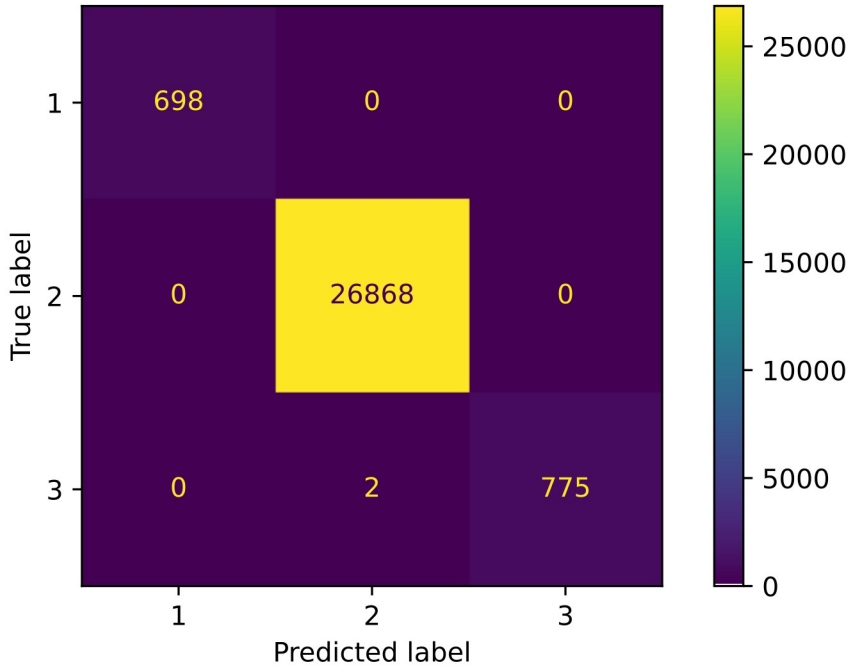
```
display_results(y_train_os, y_pred_train_os_dt, clf_os_dt)
```

```
              precision    recall  f1-score   support

           1       1.00      1.00      1.00    107467
           2       1.00      1.00      1.00    107467
           3       1.00      1.00      1.00    107467

    accuracy                           1.00    322401
   macro avg       1.00      1.00      1.00    322401
weighted avg       1.00      1.00      1.00    322401
```

In [138…

```
display_results(y_test, y_pred_test_os_dt, clf_os_dt)
```

```
              precision    recall  f1-score   support

           1       1.00      1.00      1.00       698
           2       1.00      1.00      1.00     26868
           3       1.00      1.00      1.00       777

    accuracy                           1.00     28343
   macro avg       1.00      1.00      1.00     28343
weighted avg       1.00      1.00      1.00     28343
```



5.4.2 Random forest

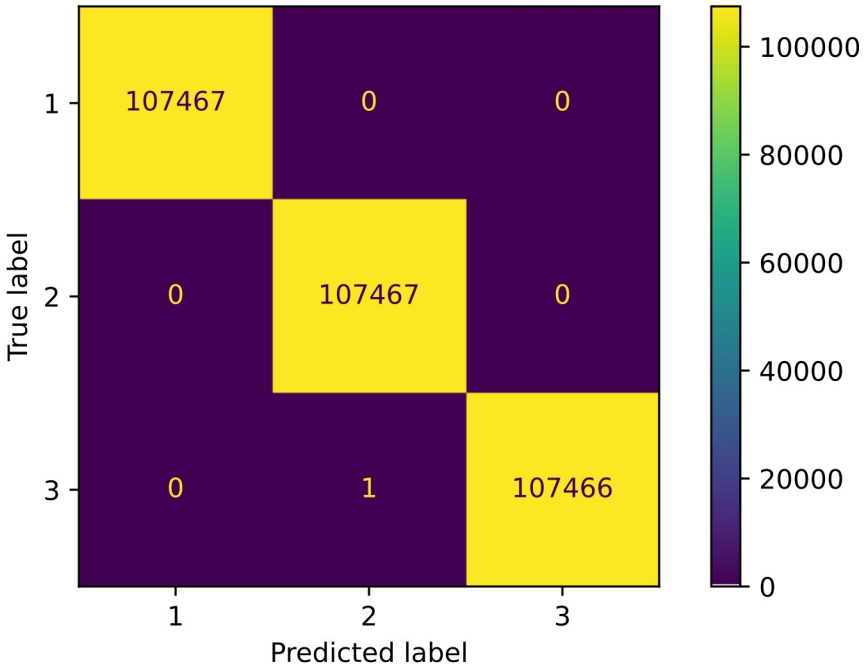
In [139…

```
y_pred_train_os_rf, y_pred_test_os_rf, clf_os_rf =clf_model(Xtrain=X_train_
          Xtest=X_test1,
          ytrain=y_train_os,
          ytest=y_test,
          classifier=RandomForestClassifier(random_state=42))
```

In [140…

```
display_results(y_train_os, y_pred_train_os_rf, clf_os_rf)
```

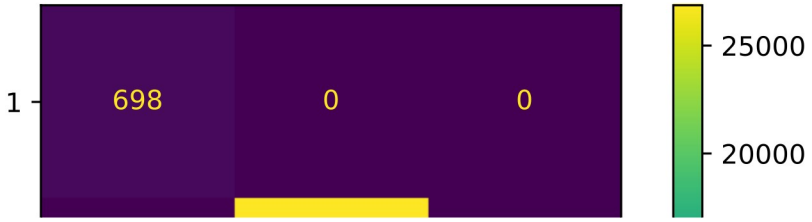
```
              precision    recall  f1-score   support

           1       1.00      1.00      1.00    107467
           2       1.00      1.00      1.00    107467
           3       1.00      1.00      1.00    107467

    accuracy                           1.00    322401
   macro avg       1.00      1.00      1.00    322401
weighted avg       1.00      1.00      1.00    322401
```



In [141…

```
display_results(y_test, y_pred_test_os_rf, clf_os_rf)
```

```
              precision    recall  f1-score   support

           1       1.00      1.00      1.00       698
           2       1.00      1.00      1.00     26868
           3       1.00      1.00      1.00       777

    accuracy                           1.00     28343
   macro avg       1.00      1.00      1.00     28343
weighted avg       1.00      1.00      1.00     28343
```



5.4.3 XGBoost

In [142…

```
y_pred_train_os_xb, y_pred_test_os_xb, clf_os_xb =clf_model(Xtrain=X_train_
          Xtest=X_test1,
          ytrain=y_train_os,
          ytest=y_test,
          classifier=XGBClassifier(random_state=42))
```

```
D:\anaconda3\lib\site-packages\xgboost\sklearn.py:1224: UserWarning: The us
e of label encoder in XGBClassifier is deprecated and will be removed in a
future release. To remove this warning, do the following: 1) Pass option us
e_label_encoder=False when constructing XGBClassifier object; and 2) Encode
your labels (y) as integers starting with 0, i.e. 0, 1, 2, ..., [num_class
- 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)
[16:17:13] WARNING: C:/Users/Administrator/workspace/xgboost-win64_release_
1.5.0/src/learner.cc:1115: Starting in XGBoost 1.3.0, the default evaluatio
n metric used with the objective 'multi:softprob' was changed from 'merror'
to 'mlogloss'. Explicitly set eval_metric if you'd like to restore the old
behavior.
```

In [143…

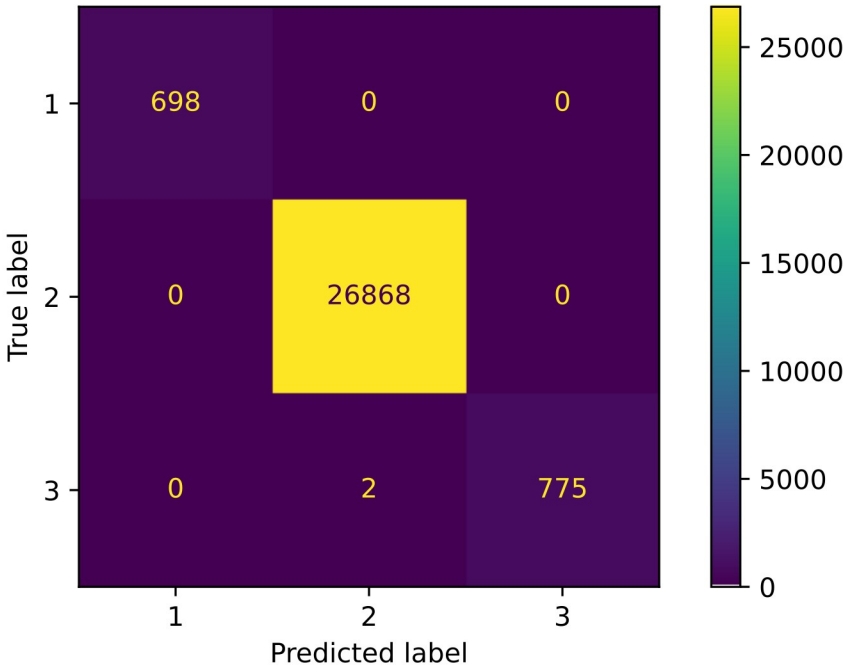
```
display_results(y_train_os, y_pred_train_os_xb, clf_os_xb)
```

```
              precision    recall  f1-score   support

           1       1.00      1.00      1.00    107467
           2       1.00      1.00      1.00    107467
           3       1.00      1.00      1.00    107467

    accuracy                           1.00    322401
   macro avg       1.00      1.00      1.00    322401
weighted avg       1.00      1.00      1.00    322401
```

In [144...

```
display_results(y_test, y_pred_test_os_xb, clf_os_xb)
```

```
              precision    recall  f1-score   support

           1       1.00      1.00      1.00       698
           2       1.00      1.00      1.00     26868
           3       1.00      1.00      1.00       777

    accuracy                           1.00     28343
   macro avg       1.00      1.00      1.00     28343
weighted avg       1.00      1.00      1.00     28343
```



5.4.4 Adaboost

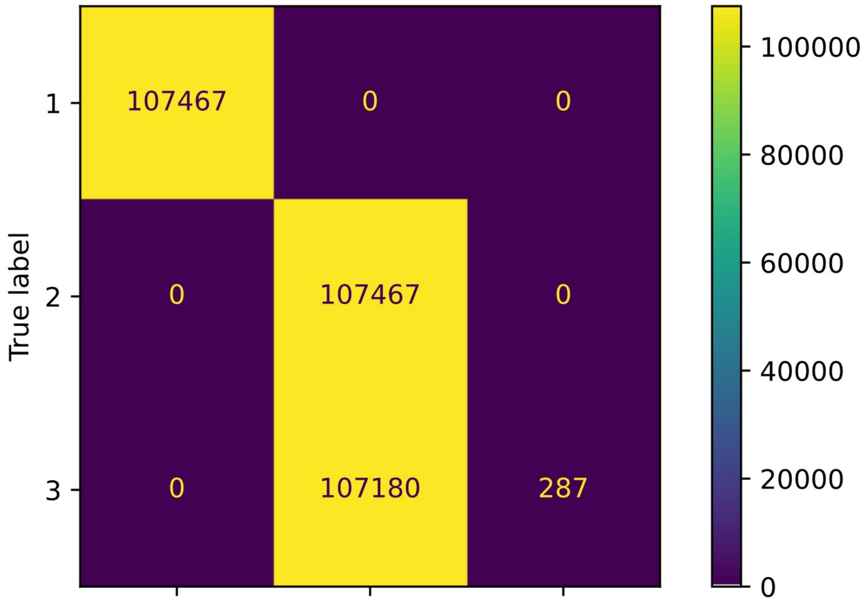
In [145...

```
y_pred_train_os_ab, y_pred_test_os_ab, clf_os_ab =clf_model(Xtrain=X_train_
          Xtest=X_test1,
          ytrain=y_train_os,
          ytest=y_test,
          classifier=AdaBoostClassifier(base_estimator=DecisionTreeClassifi
                                        random_state=42))
```

In [146...

```
display_results(y_train_os, y_pred_train_os_ab, clf_os_ab)
```

```
              precision    recall  f1-score   support

           1       1.00      1.00      1.00    107467
           2       0.50      1.00      0.67    107467
           3       1.00      0.00      0.01    107467

    accuracy                           0.67    322401
   macro avg       0.83      0.67      0.56    322401
weighted avg       0.83      0.67      0.56    322401
```



In [147…

```
display_results(y_test, y_pred_test_os_ab, clf_os_ab)
```

```
D:\anaconda3\lib\site-packages\sklearn\metrics\_classification.py:1318: Und
efinedMetricWarning: Precision and F-score are ill-defined and being set to
0.0 in labels with no predicted samples. Use `zero_division` parameter to c
ontrol this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
D:\anaconda3\lib\site-packages\sklearn\metrics\_classification.py:1318: Und
efinedMetricWarning: Precision and F-score are ill-defined and being set to
0.0 in labels with no predicted samples. Use `zero_division` parameter to c
ontrol this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
D:\anaconda3\lib\site-packages\sklearn\metrics\_classification.py:1318: Und
efinedMetricWarning: Precision and F-score are ill-defined and being set to
0.0 in labels with no predicted samples. Use `zero_division` parameter to c
ontrol this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
              precision    recall  f1-score   support

           1       1.00      1.00      1.00       698
           2       0.97      1.00      0.99     26868
           3       0.00      0.00      0.00       777

    accuracy                           0.97     28343
   macro avg       0.66      0.67      0.66     28343
weighted avg       0.95      0.97      0.96     28343
```

5.4.5 Catboost

In [148...
```
#%pip install catboost
```

In [149...
```
from catboost import CatBoostClassifier
```

In [150...
```
y_pred_train_os_cb, y_pred_test_os_cb, clf_os_cb =clf_model(Xtrain=X_train_
          Xtest=X_test1,
          ytrain=y_train_os,
          ytest=y_test,
          classifier=CatBoostClassifier(random_state=42))
```

```
Learning rate set to 0.106231
0:      learn: 0.8984739        total: 286ms    remaining: 4m 45s
1:      learn: 0.7524664        total: 403ms    remaining: 3m 20s
2:      learn: 0.6398595        total: 538ms    remaining: 2m 58s
3:      learn: 0.5499178        total: 648ms    remaining: 2m 41s
4:      learn: 0.4763763        total: 754ms    remaining: 2m 30s
5:      learn: 0.4151717        total: 867ms    remaining: 2m 23s
6:      learn: 0.3635921        total: 986ms    remaining: 2m 19s
7:      learn: 0.3196686        total: 1.1s     remaining: 2m 16s
8:      learn: 0.2819498        total: 1.22s    remaining: 2m 14s
9:      learn: 0.2493628        total: 1.33s    remaining: 2m 11s
10:     learn: 0.2210504        total: 1.44s    remaining: 2m 9s
11:     learn: 0.1963458        total: 1.56s    remaining: 2m 8s
12:     learn: 0.1747384        total: 1.67s    remaining: 2m 6s
13:     learn: 0.1557199        total: 1.79s    remaining: 2m 5s
14:     learn: 0.1389651        total: 1.9s     remaining: 2m 4s
15:     learn: 0.1241677        total: 2.02s    remaining: 2m 4s
16:     learn: 0.1110727        total: 2.15s    remaining: 2m 4s
17:     learn: 0.0994656        total: 2.27s    remaining: 2m 3s
18:     learn: 0.0891538        total: 2.39s    remaining: 2m 3s
19:     learn: 0.0800102        total: 2.51s    remaining: 2m 3s
20:     learn: 0.0718518        total: 2.64s    remaining: 2m 3s
21:     learn: 0.0645775        total: 2.76s    remaining: 2m 2s
22:     learn: 0.0580866        total: 2.88s    remaining: 2m 2s
23:     learn: 0.0522983        total: 3s       remaining: 2m 2s
24:     learn: 0.0471286        total: 3.13s    remaining: 2m 1s
25:     learn: 0.0425189        total: 3.24s    remaining: 2m 1s
26:     learn: 0.0383925        total: 3.36s    remaining: 2m 1s
27:     learn: 0.0346980        total: 3.47s    remaining: 2m
28:     learn: 0.0313889        total: 3.59s    remaining: 2m
29:     learn: 0.0284303        total: 3.7s     remaining: 1m 59s
30:     learn: 0.0257775        total: 3.81s    remaining: 1m 59s
31:     learn: 0.0233929        total: 3.92s    remaining: 1m 58s
32:     learn: 0.0212294        total: 4.03s    remaining: 1m 58s
33:     learn: 0.0191533        total: 4.14s    remaining: 1m 57s
34:     learn: 0.0174416        total: 4.26s    remaining: 1m 57s
35:     learn: 0.0159158        total: 4.38s    remaining: 1m 57s
36:     learn: 0.0143553        total: 4.5s     remaining: 1m 57s
37:     learn: 0.0130546        total: 4.61s    remaining: 1m 56s
38:     learn: 0.0118646        total: 4.72s    remaining: 1m 56s
```

```
39:     learn: 0.0108684        total: 4.84s    remaining: 1m 56s
40:     learn: 0.0099379        total: 4.95s    remaining: 1m 55s
41:     learn: 0.0091188        total: 5.08s    remaining: 1m 55s
42:     learn: 0.0083532        total: 5.19s    remaining: 1m 55s
43:     learn: 0.0076891        total: 5.32s    remaining: 1m 55s
44:     learn: 0.0070866        total: 5.43s    remaining: 1m 55s
45:     learn: 0.0065346        total: 5.55s    remaining: 1m 55s
46:     learn: 0.0060613        total: 5.67s    remaining: 1m 55s
47:     learn: 0.0056527        total: 5.78s    remaining: 1m 54s
48:     learn: 0.0052634        total: 5.89s    remaining: 1m 54s
49:     learn: 0.0049124        total: 6s       remaining: 1m 53s
50:     learn: 0.0045713        total: 6.11s    remaining: 1m 53s
51:     learn: 0.0042846        total: 6.22s    remaining: 1m 53s
52:     learn: 0.0040458        total: 6.33s    remaining: 1m 53s
53:     learn: 0.0038186        total: 6.45s    remaining: 1m 53s
54:     learn: 0.0036113        total: 6.57s    remaining: 1m 52s
55:     learn: 0.0033854        total: 6.69s    remaining: 1m 52s
56:     learn: 0.0031944        total: 6.8s     remaining: 1m 52s
57:     learn: 0.0030851        total: 6.92s    remaining: 1m 52s
58:     learn: 0.0029729        total: 7.03s    remaining: 1m 52s
59:     learn: 0.0028187        total: 7.15s    remaining: 1m 51s
60:     learn: 0.0026845        total: 7.25s    remaining: 1m 51s
61:     learn: 0.0025608        total: 7.36s    remaining: 1m 51s
62:     learn: 0.0024528        total: 7.47s    remaining: 1m 51s
63:     learn: 0.0023549        total: 7.59s    remaining: 1m 50s
64:     learn: 0.0022775        total: 7.7s     remaining: 1m 50s
65:     learn: 0.0022292        total: 7.81s    remaining: 1m 50s
66:     learn: 0.0021544        total: 7.92s    remaining: 1m 50s
67:     learn: 0.0021097        total: 8.03s    remaining: 1m 50s
68:     learn: 0.0020330        total: 8.14s    remaining: 1m 49s
69:     learn: 0.0019740        total: 8.26s    remaining: 1m 49s
70:     learn: 0.0019196        total: 8.37s    remaining: 1m 49s
71:     learn: 0.0018888        total: 8.47s    remaining: 1m 49s
72:     learn: 0.0018646        total: 8.58s    remaining: 1m 48s
73:     learn: 0.0018165        total: 8.7s     remaining: 1m 48s
74:     learn: 0.0017959        total: 8.8s     remaining: 1m 48s
75:     learn: 0.0017764        total: 8.92s    remaining: 1m 48s
76:     learn: 0.0017566        total: 9.04s    remaining: 1m 48s
77:     learn: 0.0017420        total: 9.14s    remaining: 1m 48s
78:     learn: 0.0017259        total: 9.25s    remaining: 1m 47s
79:     learn: 0.0017039        total: 9.37s    remaining: 1m 47s
80:     learn: 0.0016716        total: 9.48s    remaining: 1m 47s
81:     learn: 0.0016407        total: 9.6s     remaining: 1m 47s
82:     learn: 0.0016067        total: 9.71s    remaining: 1m 47s
83:     learn: 0.0015817        total: 9.82s    remaining: 1m 47s
84:     learn: 0.0015675        total: 9.94s    remaining: 1m 47s
85:     learn: 0.0015552        total: 10.1s    remaining: 1m 46s
86:     learn: 0.0015307        total: 10.2s    remaining: 1m 46s
87:     learn: 0.0015240        total: 10.3s    remaining: 1m 46s
88:     learn: 0.0014993        total: 10.4s    remaining: 1m 46s
89:     learn: 0.0014857        total: 10.5s    remaining: 1m 46s
90:     learn: 0.0014713        total: 10.6s    remaining: 1m 46s
91:     learn: 0.0014656        total: 10.7s    remaining: 1m 45s
92:     learn: 0.0014438        total: 10.8s    remaining: 1m 45s
93:     learn: 0.0014360        total: 11s      remaining: 1m 45s
94:     learn: 0.0014312        total: 11.1s    remaining: 1m 45s
95:     learn: 0.0014213        total: 11.2s    remaining: 1m 45s
96:     learn: 0.0014052        total: 11.3s    remaining: 1m 45s
97:     learn: 0.0013948        total: 11.4s    remaining: 1m 45s
98:     learn: 0.0013851        total: 11.5s    remaining: 1m 45s
99:     learn: 0.0013734        total: 11.7s    remaining: 1m 44s
100:    learn: 0.0013648        total: 11.8s    remaining: 1m 44s
101:    learn: 0.0013532        total: 11.9s    remaining: 1m 44s
```

```
102:    learn: 0.0013416        total: 12s      remaining: 1m 44s
103:    learn: 0.0013338        total: 12.1s    remaining: 1m 44s
104:    learn: 0.0013171        total: 12.2s    remaining: 1m 44s
105:    learn: 0.0013054        total: 12.3s    remaining: 1m 44s
106:    learn: 0.0012976        total: 12.5s    remaining: 1m 43s
107:    learn: 0.0012905        total: 12.6s    remaining: 1m 43s
108:    learn: 0.0012765        total: 12.7s    remaining: 1m 43s
109:    learn: 0.0012716        total: 12.8s    remaining: 1m 43s
110:    learn: 0.0012649        total: 12.9s    remaining: 1m 43s
111:    learn: 0.0012564        total: 13s      remaining: 1m 42s
112:    learn: 0.0012511        total: 13.1s    remaining: 1m 42s
113:    learn: 0.0012407        total: 13.2s    remaining: 1m 42s
114:    learn: 0.0012361        total: 13.3s    remaining: 1m 42s
115:    learn: 0.0012320        total: 13.4s    remaining: 1m 42s
116:    learn: 0.0012229        total: 13.5s    remaining: 1m 42s
117:    learn: 0.0012165        total: 13.6s    remaining: 1m 41s
118:    learn: 0.0012028        total: 13.8s    remaining: 1m 41s
119:    learn: 0.0012003        total: 13.9s    remaining: 1m 41s
120:    learn: 0.0011898        total: 14s      remaining: 1m 41s
121:    learn: 0.0011855        total: 14.1s    remaining: 1m 41s
122:    learn: 0.0011806        total: 14.2s    remaining: 1m 41s
123:    learn: 0.0011772        total: 14.3s    remaining: 1m 41s
124:    learn: 0.0011651        total: 14.5s    remaining: 1m 41s
125:    learn: 0.0011598        total: 14.6s    remaining: 1m 41s
126:    learn: 0.0011550        total: 14.7s    remaining: 1m 40s
127:    learn: 0.0011432        total: 14.8s    remaining: 1m 40s
128:    learn: 0.0011402        total: 14.9s    remaining: 1m 40s
129:    learn: 0.0011359        total: 15s      remaining: 1m 40s
130:    learn: 0.0011228        total: 15.1s    remaining: 1m 40s
131:    learn: 0.0011195        total: 15.2s    remaining: 1m 40s
132:    learn: 0.0011089        total: 15.3s    remaining: 1m 39s
133:    learn: 0.0011043        total: 15.4s    remaining: 1m 39s
134:    learn: 0.0010955        total: 15.6s    remaining: 1m 39s
135:    learn: 0.0010911        total: 15.7s    remaining: 1m 39s
136:    learn: 0.0010867        total: 15.8s    remaining: 1m 39s
137:    learn: 0.0010832        total: 15.9s    remaining: 1m 39s
138:    learn: 0.0010698        total: 16s      remaining: 1m 38s
139:    learn: 0.0010657        total: 16.1s    remaining: 1m 38s
140:    learn: 0.0010617        total: 16.2s    remaining: 1m 38s
141:    learn: 0.0010590        total: 16.3s    remaining: 1m 38s
142:    learn: 0.0010566        total: 16.4s    remaining: 1m 38s
143:    learn: 0.0010530        total: 16.5s    remaining: 1m 38s
144:    learn: 0.0010440        total: 16.6s    remaining: 1m 37s
145:    learn: 0.0010401        total: 16.7s    remaining: 1m 37s
146:    learn: 0.0010343        total: 16.8s    remaining: 1m 37s
147:    learn: 0.0010309        total: 16.9s    remaining: 1m 37s
148:    learn: 0.0010247        total: 17s      remaining: 1m 37s
149:    learn: 0.0010171        total: 17.2s    remaining: 1m 37s
150:    learn: 0.0010075        total: 17.3s    remaining: 1m 37s
151:    learn: 0.0010039        total: 17.4s    remaining: 1m 36s
152:    learn: 0.0009953        total: 17.5s    remaining: 1m 36s
153:    learn: 0.0009921        total: 17.6s    remaining: 1m 36s
154:    learn: 0.0009848        total: 17.7s    remaining: 1m 36s
155:    learn: 0.0009813        total: 17.8s    remaining: 1m 36s
156:    learn: 0.0009795        total: 17.9s    remaining: 1m 36s
157:    learn: 0.0009764        total: 18s      remaining: 1m 35s
158:    learn: 0.0009695        total: 18.1s    remaining: 1m 35s
159:    learn: 0.0009629        total: 18.2s    remaining: 1m 35s
160:    learn: 0.0009612        total: 18.3s    remaining: 1m 35s
161:    learn: 0.0009537        total: 18.4s    remaining: 1m 35s
162:    learn: 0.0009512        total: 18.5s    remaining: 1m 35s
163:    learn: 0.0009385        total: 18.7s    remaining: 1m 35s
164:    learn: 0.0009366        total: 18.8s    remaining: 1m 34s
```

```
165:    learn: 0.0009336        total: 18.9s    remaining: 1m 34s
166:    learn: 0.0009303        total: 19s      remaining: 1m 34s
167:    learn: 0.0009238        total: 19.1s    remaining: 1m 34s
168:    learn: 0.0009218        total: 19.2s    remaining: 1m 34s
169:    learn: 0.0009165        total: 19.3s    remaining: 1m 34s
170:    learn: 0.0009144        total: 19.4s    remaining: 1m 33s
171:    learn: 0.0009114        total: 19.5s    remaining: 1m 33s
172:    learn: 0.0009095        total: 19.6s    remaining: 1m 33s
173:    learn: 0.0009067        total: 19.7s    remaining: 1m 33s
174:    learn: 0.0009056        total: 19.8s    remaining: 1m 33s
175:    learn: 0.0009030        total: 19.9s    remaining: 1m 33s
176:    learn: 0.0008980        total: 20s      remaining: 1m 33s
177:    learn: 0.0008956        total: 20.1s    remaining: 1m 32s
178:    learn: 0.0008912        total: 20.3s    remaining: 1m 32s
179:    learn: 0.0008894        total: 20.4s    remaining: 1m 32s
180:    learn: 0.0008854        total: 20.5s    remaining: 1m 32s
181:    learn: 0.0008768        total: 20.6s    remaining: 1m 32s
182:    learn: 0.0008748        total: 20.7s    remaining: 1m 32s
183:    learn: 0.0008709        total: 20.8s    remaining: 1m 32s
184:    learn: 0.0008700        total: 20.9s    remaining: 1m 32s
185:    learn: 0.0008691        total: 21s      remaining: 1m 31s
186:    learn: 0.0008684        total: 21.1s    remaining: 1m 31s
187:    learn: 0.0008664        total: 21.2s    remaining: 1m 31s
188:    learn: 0.0008617        total: 21.3s    remaining: 1m 31s
189:    learn: 0.0008609        total: 21.4s    remaining: 1m 31s
190:    learn: 0.0008603        total: 21.6s    remaining: 1m 31s
191:    learn: 0.0008597        total: 21.7s    remaining: 1m 31s
192:    learn: 0.0008575        total: 21.8s    remaining: 1m 30s
193:    learn: 0.0008568        total: 21.9s    remaining: 1m 30s
194:    learn: 0.0008534        total: 22s      remaining: 1m 30s
195:    learn: 0.0008527        total: 22.1s    remaining: 1m 30s
196:    learn: 0.0008520        total: 22.2s    remaining: 1m 30s
197:    learn: 0.0008495        total: 22.3s    remaining: 1m 30s
198:    learn: 0.0008479        total: 22.4s    remaining: 1m 30s
199:    learn: 0.0008470        total: 22.5s    remaining: 1m 30s
200:    learn: 0.0008465        total: 22.6s    remaining: 1m 29s
201:    learn: 0.0008447        total: 22.7s    remaining: 1m 29s
202:    learn: 0.0008403        total: 22.8s    remaining: 1m 29s
203:    learn: 0.0008387        total: 22.9s    remaining: 1m 29s
204:    learn: 0.0008372        total: 23s      remaining: 1m 29s
205:    learn: 0.0008360        total: 23.1s    remaining: 1m 29s
206:    learn: 0.0008337        total: 23.3s    remaining: 1m 29s
207:    learn: 0.0008331        total: 23.4s    remaining: 1m 28s
208:    learn: 0.0008320        total: 23.5s    remaining: 1m 28s
209:    learn: 0.0008236        total: 23.6s    remaining: 1m 28s
210:    learn: 0.0008226        total: 23.7s    remaining: 1m 28s
211:    learn: 0.0008221        total: 23.8s    remaining: 1m 28s
212:    learn: 0.0008214        total: 23.9s    remaining: 1m 28s
213:    learn: 0.0008165        total: 24s      remaining: 1m 28s
214:    learn: 0.0008160        total: 24.1s    remaining: 1m 28s
215:    learn: 0.0008149        total: 24.2s    remaining: 1m 27s
216:    learn: 0.0008134        total: 24.3s    remaining: 1m 27s
217:    learn: 0.0008129        total: 24.4s    remaining: 1m 27s
218:    learn: 0.0008118        total: 24.6s    remaining: 1m 27s
219:    learn: 0.0008082        total: 24.7s    remaining: 1m 27s
220:    learn: 0.0008079        total: 24.8s    remaining: 1m 27s
221:    learn: 0.0008074        total: 24.9s    remaining: 1m 27s
222:    learn: 0.0008062        total: 25s      remaining: 1m 27s
223:    learn: 0.0008053        total: 25.1s    remaining: 1m 26s
224:    learn: 0.0008034        total: 25.2s    remaining: 1m 26s
225:    learn: 0.0008028        total: 25.3s    remaining: 1m 26s
226:    learn: 0.0007999        total: 25.4s    remaining: 1m 26s
227:    learn: 0.0007991        total: 25.5s    remaining: 1m 26s
```

```
228:    learn: 0.0007982        total: 25.6s    remaining: 1m 26s
229:    learn: 0.0007979        total: 25.8s    remaining: 1m 26s
230:    learn: 0.0007914        total: 25.9s    remaining: 1m 26s
231:    learn: 0.0007883        total: 26s      remaining: 1m 25s
232:    learn: 0.0007876        total: 26.1s    remaining: 1m 25s
233:    learn: 0.0007863        total: 26.2s    remaining: 1m 25s
234:    learn: 0.0007859        total: 26.3s    remaining: 1m 25s
235:    learn: 0.0007802        total: 26.4s    remaining: 1m 25s
236:    learn: 0.0007793        total: 26.5s    remaining: 1m 25s
237:    learn: 0.0007786        total: 26.6s    remaining: 1m 25s
238:    learn: 0.0007734        total: 26.7s    remaining: 1m 25s
239:    learn: 0.0007726        total: 26.9s    remaining: 1m 25s
240:    learn: 0.0007716        total: 27s      remaining: 1m 24s
241:    learn: 0.0007712        total: 27.1s    remaining: 1m 24s
242:    learn: 0.0007690        total: 27.2s    remaining: 1m 24s
243:    learn: 0.0007685        total: 27.3s    remaining: 1m 24s
244:    learn: 0.0007676        total: 27.4s    remaining: 1m 24s
245:    learn: 0.0007648        total: 27.5s    remaining: 1m 24s
246:    learn: 0.0007645        total: 27.6s    remaining: 1m 24s
247:    learn: 0.0007638        total: 27.7s    remaining: 1m 24s
248:    learn: 0.0007633        total: 27.8s    remaining: 1m 23s
249:    learn: 0.0007626        total: 27.9s    remaining: 1m 23s
250:    learn: 0.0007622        total: 28s      remaining: 1m 23s
251:    learn: 0.0007616        total: 28.1s    remaining: 1m 23s
252:    learn: 0.0007614        total: 28.3s    remaining: 1m 23s
253:    learn: 0.0007608        total: 28.4s    remaining: 1m 23s
254:    learn: 0.0007604        total: 28.5s    remaining: 1m 23s
255:    learn: 0.0007556        total: 28.6s    remaining: 1m 23s
256:    learn: 0.0007552        total: 28.7s    remaining: 1m 22s
257:    learn: 0.0007526        total: 28.8s    remaining: 1m 22s
258:    learn: 0.0007522        total: 28.9s    remaining: 1m 22s
259:    learn: 0.0007473        total: 29s      remaining: 1m 22s
260:    learn: 0.0007405        total: 29.1s    remaining: 1m 22s
261:    learn: 0.0007357        total: 29.2s    remaining: 1m 22s
262:    learn: 0.0007344        total: 29.3s    remaining: 1m 22s
263:    learn: 0.0007341        total: 29.4s    remaining: 1m 22s
264:    learn: 0.0007330        total: 29.5s    remaining: 1m 21s
265:    learn: 0.0007297        total: 29.6s    remaining: 1m 21s
266:    learn: 0.0007238        total: 29.8s    remaining: 1m 21s
267:    learn: 0.0007235        total: 29.9s    remaining: 1m 21s
268:    learn: 0.0007199        total: 30s      remaining: 1m 21s
269:    learn: 0.0007196        total: 30.1s    remaining: 1m 21s
270:    learn: 0.0007194        total: 30.2s    remaining: 1m 21s
271:    learn: 0.0007159        total: 30.3s    remaining: 1m 21s
272:    learn: 0.0007157        total: 30.4s    remaining: 1m 20s
273:    learn: 0.0007146        total: 30.5s    remaining: 1m 20s
274:    learn: 0.0007141        total: 30.6s    remaining: 1m 20s
275:    learn: 0.0007139        total: 30.7s    remaining: 1m 20s
276:    learn: 0.0007104        total: 30.8s    remaining: 1m 20s
277:    learn: 0.0007103        total: 30.9s    remaining: 1m 20s
278:    learn: 0.0007100        total: 31s      remaining: 1m 20s
279:    learn: 0.0007089        total: 31.1s    remaining: 1m 20s
280:    learn: 0.0007079        total: 31.2s    remaining: 1m 19s
281:    learn: 0.0007027        total: 31.3s    remaining: 1m 19s
282:    learn: 0.0006984        total: 31.5s    remaining: 1m 19s
283:    learn: 0.0006938        total: 31.6s    remaining: 1m 19s
284:    learn: 0.0006926        total: 31.7s    remaining: 1m 19s
285:    learn: 0.0006902        total: 31.8s    remaining: 1m 19s
286:    learn: 0.0006896        total: 31.9s    remaining: 1m 19s
287:    learn: 0.0006894        total: 32s      remaining: 1m 19s
288:    learn: 0.0006889        total: 32.1s    remaining: 1m 18s
289:    learn: 0.0006883        total: 32.2s    remaining: 1m 18s
290:    learn: 0.0006862        total: 32.3s    remaining: 1m 18s
```

```
291:    learn: 0.0006854        total: 32.4s    remaining: 1m 18s
292:    learn: 0.0006839        total: 32.5s    remaining: 1m 18s
293:    learn: 0.0006833        total: 32.6s    remaining: 1m 18s
294:    learn: 0.0006831        total: 32.8s    remaining: 1m 18s
295:    learn: 0.0006827        total: 32.9s    remaining: 1m 18s
296:    learn: 0.0006817        total: 33s      remaining: 1m 18s
297:    learn: 0.0006813        total: 33.1s    remaining: 1m 17s
298:    learn: 0.0006803        total: 33.2s    remaining: 1m 17s
299:    learn: 0.0006782        total: 33.3s    remaining: 1m 17s
300:    learn: 0.0006779        total: 33.4s    remaining: 1m 17s
301:    learn: 0.0006776        total: 33.5s    remaining: 1m 17s
302:    learn: 0.0006772        total: 33.6s    remaining: 1m 17s
303:    learn: 0.0006770        total: 33.7s    remaining: 1m 17s
304:    learn: 0.0006766        total: 33.8s    remaining: 1m 17s
305:    learn: 0.0006763        total: 33.9s    remaining: 1m 16s
306:    learn: 0.0006761        total: 34s      remaining: 1m 16s
307:    learn: 0.0006747        total: 34.1s    remaining: 1m 16s
308:    learn: 0.0006707        total: 34.2s    remaining: 1m 16s
309:    learn: 0.0006704        total: 34.3s    remaining: 1m 16s
310:    learn: 0.0006678        total: 34.4s    remaining: 1m 16s
311:    learn: 0.0006666        total: 34.5s    remaining: 1m 16s
312:    learn: 0.0006661        total: 34.6s    remaining: 1m 16s
313:    learn: 0.0006618        total: 34.8s    remaining: 1m 15s
314:    learn: 0.0006616        total: 34.9s    remaining: 1m 15s
315:    learn: 0.0006568        total: 35s      remaining: 1m 15s
316:    learn: 0.0006567        total: 35.1s    remaining: 1m 15s
317:    learn: 0.0006565        total: 35.2s    remaining: 1m 15s
318:    learn: 0.0006562        total: 35.3s    remaining: 1m 15s
319:    learn: 0.0006559        total: 35.4s    remaining: 1m 15s
320:    learn: 0.0006557        total: 35.5s    remaining: 1m 15s
321:    learn: 0.0006555        total: 35.6s    remaining: 1m 15s
322:    learn: 0.0006555        total: 35.7s    remaining: 1m 14s
323:    learn: 0.0006554        total: 35.8s    remaining: 1m 14s
324:    learn: 0.0006553        total: 35.9s    remaining: 1m 14s
325:    learn: 0.0006545        total: 36s      remaining: 1m 14s
326:    learn: 0.0006544        total: 36.1s    remaining: 1m 14s
327:    learn: 0.0006543        total: 36.2s    remaining: 1m 14s
328:    learn: 0.0006535        total: 36.3s    remaining: 1m 14s
329:    learn: 0.0006534        total: 36.5s    remaining: 1m 14s
330:    learn: 0.0006532        total: 36.6s    remaining: 1m 13s
331:    learn: 0.0006529        total: 36.7s    remaining: 1m 13s
332:    learn: 0.0006488        total: 36.8s    remaining: 1m 13s
333:    learn: 0.0006462        total: 36.9s    remaining: 1m 13s
334:    learn: 0.0006426        total: 37s      remaining: 1m 13s
335:    learn: 0.0006421        total: 37.1s    remaining: 1m 13s
336:    learn: 0.0006420        total: 37.2s    remaining: 1m 13s
337:    learn: 0.0006417        total: 37.3s    remaining: 1m 13s
338:    learn: 0.0006407        total: 37.4s    remaining: 1m 12s
339:    learn: 0.0006366        total: 37.5s    remaining: 1m 12s
340:    learn: 0.0006365        total: 37.7s    remaining: 1m 12s
341:    learn: 0.0006354        total: 37.8s    remaining: 1m 12s
342:    learn: 0.0006316        total: 37.9s    remaining: 1m 12s
343:    learn: 0.0006314        total: 38s      remaining: 1m 12s
344:    learn: 0.0006304        total: 38.1s    remaining: 1m 12s
345:    learn: 0.0006267        total: 38.2s    remaining: 1m 12s
346:    learn: 0.0006264        total: 38.3s    remaining: 1m 12s
347:    learn: 0.0006263        total: 38.4s    remaining: 1m 11s
348:    learn: 0.0006261        total: 38.5s    remaining: 1m 11s
349:    learn: 0.0006260        total: 38.6s    remaining: 1m 11s
350:    learn: 0.0006258        total: 38.7s    remaining: 1m 11s
351:    learn: 0.0006258        total: 38.9s    remaining: 1m 11s
352:    learn: 0.0006251        total: 39s      remaining: 1m 11s
353:    learn: 0.0006249        total: 39.1s    remaining: 1m 11s
```

```
354:    learn: 0.0006243        total: 39.2s    remaining: 1m 11s
355:    learn: 0.0006241        total: 39.3s    remaining: 1m 11s
356:    learn: 0.0006241        total: 39.4s    remaining: 1m 10s
357:    learn: 0.0006209        total: 39.5s    remaining: 1m 10s
358:    learn: 0.0006208        total: 39.6s    remaining: 1m 10s
359:    learn: 0.0006207        total: 39.7s    remaining: 1m 10s
360:    learn: 0.0006206        total: 39.8s    remaining: 1m 10s
361:    learn: 0.0006205        total: 39.9s    remaining: 1m 10s
362:    learn: 0.0006205        total: 40s      remaining: 1m 10s
363:    learn: 0.0006204        total: 40.1s    remaining: 1m 10s
364:    learn: 0.0006204        total: 40.2s    remaining: 1m 9s
365:    learn: 0.0006170        total: 40.3s    remaining: 1m 9s
366:    learn: 0.0006169        total: 40.4s    remaining: 1m 9s
367:    learn: 0.0006168        total: 40.5s    remaining: 1m 9s
368:    learn: 0.0006167        total: 40.6s    remaining: 1m 9s
369:    learn: 0.0006165        total: 40.7s    remaining: 1m 9s
370:    learn: 0.0006165        total: 40.9s    remaining: 1m 9s
371:    learn: 0.0006138        total: 41s      remaining: 1m 9s
372:    learn: 0.0006136        total: 41.1s    remaining: 1m 9s
373:    learn: 0.0006116        total: 41.2s    remaining: 1m 8s
374:    learn: 0.0006090        total: 41.3s    remaining: 1m 8s
375:    learn: 0.0006086        total: 41.4s    remaining: 1m 8s
376:    learn: 0.0006070        total: 41.5s    remaining: 1m 8s
377:    learn: 0.0006069        total: 41.6s    remaining: 1m 8s
378:    learn: 0.0006068        total: 41.7s    remaining: 1m 8s
379:    learn: 0.0006036        total: 41.8s    remaining: 1m 8s
380:    learn: 0.0006019        total: 41.9s    remaining: 1m 8s
381:    learn: 0.0006018        total: 42s      remaining: 1m 8s
382:    learn: 0.0006016        total: 42.1s    remaining: 1m 7s
383:    learn: 0.0005985        total: 42.2s    remaining: 1m 7s
384:    learn: 0.0005983        total: 42.4s    remaining: 1m 7s
385:    learn: 0.0005982        total: 42.5s    remaining: 1m 7s
386:    learn: 0.0005981        total: 42.6s    remaining: 1m 7s
387:    learn: 0.0005971        total: 42.7s    remaining: 1m 7s
388:    learn: 0.0005944        total: 42.8s    remaining: 1m 7s
389:    learn: 0.0005943        total: 42.9s    remaining: 1m 7s
390:    learn: 0.0005942        total: 43s      remaining: 1m 6s
391:    learn: 0.0005941        total: 43.1s    remaining: 1m 6s
392:    learn: 0.0005938        total: 43.2s    remaining: 1m 6s
393:    learn: 0.0005917        total: 43.3s    remaining: 1m 6s
394:    learn: 0.0005912        total: 43.5s    remaining: 1m 6s
395:    learn: 0.0005912        total: 43.6s    remaining: 1m 6s
396:    learn: 0.0005904        total: 43.7s    remaining: 1m 6s
397:    learn: 0.0005903        total: 43.8s    remaining: 1m 6s
398:    learn: 0.0005902        total: 43.9s    remaining: 1m 6s
399:    learn: 0.0005902        total: 44s      remaining: 1m 5s
400:    learn: 0.0005901        total: 44.1s    remaining: 1m 5s
401:    learn: 0.0005899        total: 44.2s    remaining: 1m 5s
402:    learn: 0.0005880        total: 44.3s    remaining: 1m 5s
403:    learn: 0.0005879        total: 44.4s    remaining: 1m 5s
404:    learn: 0.0005877        total: 44.5s    remaining: 1m 5s
405:    learn: 0.0005848        total: 44.6s    remaining: 1m 5s
406:    learn: 0.0005842        total: 44.7s    remaining: 1m 5s
407:    learn: 0.0005838        total: 44.8s    remaining: 1m 5s
408:    learn: 0.0005836        total: 45s      remaining: 1m 4s
409:    learn: 0.0005835        total: 45.1s    remaining: 1m 4s
410:    learn: 0.0005829        total: 45.2s    remaining: 1m 4s
411:    learn: 0.0005825        total: 45.3s    remaining: 1m 4s
412:    learn: 0.0005806        total: 45.4s    remaining: 1m 4s
413:    learn: 0.0005805        total: 45.5s    remaining: 1m 4s
414:    learn: 0.0005803        total: 45.6s    remaining: 1m 4s
415:    learn: 0.0005791        total: 45.7s    remaining: 1m 4s
416:    learn: 0.0005788        total: 45.8s    remaining: 1m 4s
```

```
417:    learn: 0.0005787        total: 45.9s    remaining: 1m 3s
418:    learn: 0.0005786        total: 46s      remaining: 1m 3s
419:    learn: 0.0005784        total: 46.1s    remaining: 1m 3s
420:    learn: 0.0005783        total: 46.2s    remaining: 1m 3s
421:    learn: 0.0005782        total: 46.3s    remaining: 1m 3s
422:    learn: 0.0005781        total: 46.5s    remaining: 1m 3s
423:    learn: 0.0005779        total: 46.6s    remaining: 1m 3s
424:    learn: 0.0005778        total: 46.7s    remaining: 1m 3s
425:    learn: 0.0005778        total: 46.8s    remaining: 1m 3s
426:    learn: 0.0005778        total: 46.9s    remaining: 1m 2s
427:    learn: 0.0005777        total: 47s      remaining: 1m 2s
428:    learn: 0.0005776        total: 47.1s    remaining: 1m 2s
429:    learn: 0.0005775        total: 47.2s    remaining: 1m 2s
430:    learn: 0.0005775        total: 47.3s    remaining: 1m 2s
431:    learn: 0.0005773        total: 47.4s    remaining: 1m 2s
432:    learn: 0.0005772        total: 47.5s    remaining: 1m 2s
433:    learn: 0.0005771        total: 47.6s    remaining: 1m 2s
434:    learn: 0.0005735        total: 47.7s    remaining: 1m 1s
435:    learn: 0.0005733        total: 47.8s    remaining: 1m 1s
436:    learn: 0.0005701        total: 47.9s    remaining: 1m 1s
437:    learn: 0.0005701        total: 48s      remaining: 1m 1s
438:    learn: 0.0005699        total: 48.1s    remaining: 1m 1s
439:    learn: 0.0005698        total: 48.3s    remaining: 1m 1s
440:    learn: 0.0005691        total: 48.3s    remaining: 1m 1s
441:    learn: 0.0005690        total: 48.5s    remaining: 1m 1s
442:    learn: 0.0005690        total: 48.6s    remaining: 1m 1s
443:    learn: 0.0005690        total: 48.7s    remaining: 1m
444:    learn: 0.0005687        total: 48.8s    remaining: 1m
445:    learn: 0.0005687        total: 48.9s    remaining: 1m
446:    learn: 0.0005686        total: 49s      remaining: 1m
447:    learn: 0.0005686        total: 49.1s    remaining: 1m
448:    learn: 0.0005686        total: 49.2s    remaining: 1m
449:    learn: 0.0005685        total: 49.3s    remaining: 1m
450:    learn: 0.0005684        total: 49.4s    remaining: 1m
451:    learn: 0.0005683        total: 49.5s    remaining: 1m
452:    learn: 0.0005682        total: 49.6s    remaining: 59.9s
453:    learn: 0.0005681        total: 49.7s    remaining: 59.8s
454:    learn: 0.0005680        total: 49.8s    remaining: 59.7s
455:    learn: 0.0005680        total: 49.9s    remaining: 59.6s
456:    learn: 0.0005678        total: 50s      remaining: 59.4s
457:    learn: 0.0005660        total: 50.1s    remaining: 59.3s
458:    learn: 0.0005658        total: 50.2s    remaining: 59.2s
459:    learn: 0.0005638        total: 50.3s    remaining: 59.1s
460:    learn: 0.0005638        total: 50.4s    remaining: 59s
461:    learn: 0.0005636        total: 50.6s    remaining: 58.9s
462:    learn: 0.0005635        total: 50.7s    remaining: 58.8s
463:    learn: 0.0005635        total: 50.8s    remaining: 58.7s
464:    learn: 0.0005620        total: 50.9s    remaining: 58.6s
465:    learn: 0.0005617        total: 51s      remaining: 58.4s
466:    learn: 0.0005615        total: 51.1s    remaining: 58.3s
467:    learn: 0.0005613        total: 51.2s    remaining: 58.2s
468:    learn: 0.0005595        total: 51.3s    remaining: 58.1s
469:    learn: 0.0005570        total: 51.4s    remaining: 58s
470:    learn: 0.0005570        total: 51.5s    remaining: 57.9s
471:    learn: 0.0005569        total: 51.6s    remaining: 57.8s
472:    learn: 0.0005569        total: 51.7s    remaining: 57.7s
473:    learn: 0.0005568        total: 51.8s    remaining: 57.5s
474:    learn: 0.0005568        total: 52s      remaining: 57.4s
475:    learn: 0.0005566        total: 52.1s    remaining: 57.3s
476:    learn: 0.0005565        total: 52.2s    remaining: 57.2s
477:    learn: 0.0005564        total: 52.3s    remaining: 57.1s
478:    learn: 0.0005563        total: 52.4s    remaining: 57s
479:    learn: 0.0005563        total: 52.5s    remaining: 56.9s
```

```
480:    learn: 0.0005563        total: 52.6s    remaining: 56.7s
481:    learn: 0.0005561        total: 52.7s    remaining: 56.6s
482:    learn: 0.0005561        total: 52.8s    remaining: 56.5s
483:    learn: 0.0005561        total: 52.9s    remaining: 56.4s
484:    learn: 0.0005555        total: 53s      remaining: 56.3s
485:    learn: 0.0005545        total: 53.1s    remaining: 56.2s
486:    learn: 0.0005544        total: 53.2s    remaining: 56.1s
487:    learn: 0.0005541        total: 53.3s    remaining: 55.9s
488:    learn: 0.0005540        total: 53.4s    remaining: 55.8s
489:    learn: 0.0005526        total: 53.5s    remaining: 55.7s
490:    learn: 0.0005526        total: 53.6s    remaining: 55.6s
491:    learn: 0.0005525        total: 53.8s    remaining: 55.5s
492:    learn: 0.0005503        total: 53.9s    remaining: 55.4s
493:    learn: 0.0005487        total: 54s      remaining: 55.3s
494:    learn: 0.0005486        total: 54.1s    remaining: 55.2s
495:    learn: 0.0005486        total: 54.2s    remaining: 55.1s
496:    learn: 0.0005485        total: 54.3s    remaining: 55s
497:    learn: 0.0005485        total: 54.4s    remaining: 54.8s
498:    learn: 0.0005482        total: 54.5s    remaining: 54.7s
499:    learn: 0.0005482        total: 54.6s    remaining: 54.6s
500:    learn: 0.0005482        total: 54.7s    remaining: 54.5s
501:    learn: 0.0005481        total: 54.8s    remaining: 54.4s
502:    learn: 0.0005450        total: 54.9s    remaining: 54.3s
503:    learn: 0.0005429        total: 55s      remaining: 54.2s
504:    learn: 0.0005428        total: 55.1s    remaining: 54s
505:    learn: 0.0005419        total: 55.2s    remaining: 53.9s
506:    learn: 0.0005413        total: 55.3s    remaining: 53.8s
507:    learn: 0.0005412        total: 55.4s    remaining: 53.7s
508:    learn: 0.0005408        total: 55.5s    remaining: 53.6s
509:    learn: 0.0005408        total: 55.7s    remaining: 53.5s
510:    learn: 0.0005407        total: 55.8s    remaining: 53.4s
511:    learn: 0.0005407        total: 55.9s    remaining: 53.2s
512:    learn: 0.0005407        total: 56s      remaining: 53.1s
513:    learn: 0.0005405        total: 56.1s    remaining: 53s
514:    learn: 0.0005388        total: 56.2s    remaining: 52.9s
515:    learn: 0.0005387        total: 56.3s    remaining: 52.8s
516:    learn: 0.0005385        total: 56.4s    remaining: 52.7s
517:    learn: 0.0005384        total: 56.5s    remaining: 52.6s
518:    learn: 0.0005384        total: 56.6s    remaining: 52.5s
519:    learn: 0.0005384        total: 56.7s    remaining: 52.3s
520:    learn: 0.0005383        total: 56.8s    remaining: 52.2s
521:    learn: 0.0005382        total: 56.9s    remaining: 52.1s
522:    learn: 0.0005370        total: 57s      remaining: 52s
523:    learn: 0.0005370        total: 57.1s    remaining: 51.9s
524:    learn: 0.0005370        total: 57.3s    remaining: 51.8s
525:    learn: 0.0005369        total: 57.4s    remaining: 51.7s
526:    learn: 0.0005338        total: 57.5s    remaining: 51.6s
527:    learn: 0.0005337        total: 57.6s    remaining: 51.5s
528:    learn: 0.0005336        total: 57.7s    remaining: 51.4s
529:    learn: 0.0005336        total: 57.8s    remaining: 51.2s
530:    learn: 0.0005335        total: 57.9s    remaining: 51.1s
531:    learn: 0.0005326        total: 58s      remaining: 51s
532:    learn: 0.0005307        total: 58.1s    remaining: 50.9s
533:    learn: 0.0005307        total: 58.2s    remaining: 50.8s
534:    learn: 0.0005305        total: 58.3s    remaining: 50.7s
535:    learn: 0.0005304        total: 58.4s    remaining: 50.6s
536:    learn: 0.0005304        total: 58.5s    remaining: 50.5s
537:    learn: 0.0005303        total: 58.6s    remaining: 50.3s
538:    learn: 0.0005293        total: 58.7s    remaining: 50.2s
539:    learn: 0.0005292        total: 58.8s    remaining: 50.1s
540:    learn: 0.0005292        total: 58.9s    remaining: 50s
541:    learn: 0.0005279        total: 59s      remaining: 49.9s
542:    learn: 0.0005278        total: 59.2s    remaining: 49.8s
```

```
543:    learn: 0.0005278        total: 59.3s    remaining: 49.7s
544:    learn: 0.0005266        total: 59.4s    remaining: 49.6s
545:    learn: 0.0005264        total: 59.5s    remaining: 49.4s
546:    learn: 0.0005235        total: 59.6s    remaining: 49.3s
547:    learn: 0.0005235        total: 59.7s    remaining: 49.2s
548:    learn: 0.0005234        total: 59.8s    remaining: 49.1s
549:    learn: 0.0005234        total: 59.9s    remaining: 49s
550:    learn: 0.0005215        total: 1m       remaining: 48.9s
551:    learn: 0.0005215        total: 1m       remaining: 48.8s
552:    learn: 0.0005215        total: 1m       remaining: 48.7s
553:    learn: 0.0005211        total: 1m       remaining: 48.6s
554:    learn: 0.0005210        total: 1m       remaining: 48.5s
555:    learn: 0.0005210        total: 1m       remaining: 48.3s
556:    learn: 0.0005210        total: 1m       remaining: 48.2s
557:    learn: 0.0005208        total: 1m       remaining: 48.1s
558:    learn: 0.0005208        total: 1m       remaining: 48s
559:    learn: 0.0005197        total: 1m       remaining: 47.9s
560:    learn: 0.0005197        total: 1m 1s    remaining: 47.8s
561:    learn: 0.0005197        total: 1m 1s    remaining: 47.7s
562:    learn: 0.0005196        total: 1m 1s    remaining: 47.6s
563:    learn: 0.0005196        total: 1m 1s    remaining: 47.5s
564:    learn: 0.0005196        total: 1m 1s    remaining: 47.4s
565:    learn: 0.0005195        total: 1m 1s    remaining: 47.3s
566:    learn: 0.0005194        total: 1m 1s    remaining: 47.1s
567:    learn: 0.0005194        total: 1m 1s    remaining: 47s
568:    learn: 0.0005194        total: 1m 1s    remaining: 46.9s
569:    learn: 0.0005184        total: 1m 2s    remaining: 46.8s
570:    learn: 0.0005158        total: 1m 2s    remaining: 46.7s
571:    learn: 0.0005143        total: 1m 2s    remaining: 46.6s
572:    learn: 0.0005142        total: 1m 2s    remaining: 46.5s
573:    learn: 0.0005142        total: 1m 2s    remaining: 46.4s
574:    learn: 0.0005142        total: 1m 2s    remaining: 46.2s
575:    learn: 0.0005142        total: 1m 2s    remaining: 46.1s
576:    learn: 0.0005141        total: 1m 2s    remaining: 46s
577:    learn: 0.0005138        total: 1m 2s    remaining: 45.9s
578:    learn: 0.0005114        total: 1m 2s    remaining: 45.8s
579:    learn: 0.0005113        total: 1m 3s    remaining: 45.7s
580:    learn: 0.0005112        total: 1m 3s    remaining: 45.6s
581:    learn: 0.0005093        total: 1m 3s    remaining: 45.5s
582:    learn: 0.0005093        total: 1m 3s    remaining: 45.4s
583:    learn: 0.0005071        total: 1m 3s    remaining: 45.3s
584:    learn: 0.0005071        total: 1m 3s    remaining: 45.1s
585:    learn: 0.0005071        total: 1m 3s    remaining: 45s
586:    learn: 0.0005071        total: 1m 3s    remaining: 44.9s
587:    learn: 0.0005070        total: 1m 3s    remaining: 44.8s
588:    learn: 0.0005069        total: 1m 4s    remaining: 44.7s
589:    learn: 0.0005068        total: 1m 4s    remaining: 44.6s
590:    learn: 0.0005068        total: 1m 4s    remaining: 44.5s
591:    learn: 0.0005050        total: 1m 4s    remaining: 44.4s
592:    learn: 0.0005047        total: 1m 4s    remaining: 44.3s
593:    learn: 0.0005036        total: 1m 4s    remaining: 44.2s
594:    learn: 0.0005019        total: 1m 4s    remaining: 44.1s
595:    learn: 0.0005019        total: 1m 4s    remaining: 44s
596:    learn: 0.0005018        total: 1m 4s    remaining: 43.8s
597:    learn: 0.0005011        total: 1m 5s    remaining: 43.7s
598:    learn: 0.0005011        total: 1m 5s    remaining: 43.6s
599:    learn: 0.0005009        total: 1m 5s    remaining: 43.5s
600:    learn: 0.0005008        total: 1m 5s    remaining: 43.4s
601:    learn: 0.0005000        total: 1m 5s    remaining: 43.3s
602:    learn: 0.0004992        total: 1m 5s    remaining: 43.2s
603:    learn: 0.0004976        total: 1m 5s    remaining: 43.1s
604:    learn: 0.0004976        total: 1m 5s    remaining: 43s
605:    learn: 0.0004975        total: 1m 5s    remaining: 42.9s
```

```
606:    learn: 0.0004975        total: 1m 6s    remaining: 42.8s
607:    learn: 0.0004974        total: 1m 6s    remaining: 42.7s
608:    learn: 0.0004967        total: 1m 6s    remaining: 42.5s
609:    learn: 0.0004957        total: 1m 6s    remaining: 42.4s
610:    learn: 0.0004956        total: 1m 6s    remaining: 42.3s
611:    learn: 0.0004956        total: 1m 6s    remaining: 42.2s
612:    learn: 0.0004955        total: 1m 6s    remaining: 42.1s
613:    learn: 0.0004953        total: 1m 6s    remaining: 42s
614:    learn: 0.0004943        total: 1m 6s    remaining: 41.9s
615:    learn: 0.0004941        total: 1m 7s    remaining: 41.8s
616:    learn: 0.0004940        total: 1m 7s    remaining: 41.7s
617:    learn: 0.0004940        total: 1m 7s    remaining: 41.5s
618:    learn: 0.0004935        total: 1m 7s    remaining: 41.4s
619:    learn: 0.0004934        total: 1m 7s    remaining: 41.3s
620:    learn: 0.0004927        total: 1m 7s    remaining: 41.2s
621:    learn: 0.0004927        total: 1m 7s    remaining: 41.1s
622:    learn: 0.0004927        total: 1m 7s    remaining: 41s
623:    learn: 0.0004926        total: 1m 7s    remaining: 40.9s
624:    learn: 0.0004926        total: 1m 7s    remaining: 40.8s
625:    learn: 0.0004926        total: 1m 8s    remaining: 40.7s
626:    learn: 0.0004926        total: 1m 8s    remaining: 40.6s
627:    learn: 0.0004924        total: 1m 8s    remaining: 40.4s
628:    learn: 0.0004924        total: 1m 8s    remaining: 40.3s
629:    learn: 0.0004924        total: 1m 8s    remaining: 40.2s
630:    learn: 0.0004923        total: 1m 8s    remaining: 40.1s
631:    learn: 0.0004923        total: 1m 8s    remaining: 40s
632:    learn: 0.0004915        total: 1m 8s    remaining: 39.9s
633:    learn: 0.0004913        total: 1m 8s    remaining: 39.8s
634:    learn: 0.0004913        total: 1m 9s    remaining: 39.7s
635:    learn: 0.0004909        total: 1m 9s    remaining: 39.6s
636:    learn: 0.0004907        total: 1m 9s    remaining: 39.4s
637:    learn: 0.0004891        total: 1m 9s    remaining: 39.3s
638:    learn: 0.0004889        total: 1m 9s    remaining: 39.2s
639:    learn: 0.0004889        total: 1m 9s    remaining: 39.1s
640:    learn: 0.0004888        total: 1m 9s    remaining: 39s
641:    learn: 0.0004886        total: 1m 9s    remaining: 38.9s
642:    learn: 0.0004886        total: 1m 9s    remaining: 38.8s
643:    learn: 0.0004886        total: 1m 9s    remaining: 38.7s
644:    learn: 0.0004883        total: 1m 10s   remaining: 38.6s
645:    learn: 0.0004883        total: 1m 10s   remaining: 38.5s
646:    learn: 0.0004881        total: 1m 10s   remaining: 38.3s
647:    learn: 0.0004879        total: 1m 10s   remaining: 38.2s
648:    learn: 0.0004873        total: 1m 10s   remaining: 38.1s
649:    learn: 0.0004863        total: 1m 10s   remaining: 38s
650:    learn: 0.0004862        total: 1m 10s   remaining: 37.9s
651:    learn: 0.0004862        total: 1m 10s   remaining: 37.8s
652:    learn: 0.0004862        total: 1m 10s   remaining: 37.7s
653:    learn: 0.0004862        total: 1m 11s   remaining: 37.6s
654:    learn: 0.0004862        total: 1m 11s   remaining: 37.5s
655:    learn: 0.0004853        total: 1m 11s   remaining: 37.3s
656:    learn: 0.0004831        total: 1m 11s   remaining: 37.2s
657:    learn: 0.0004831        total: 1m 11s   remaining: 37.1s
658:    learn: 0.0004825        total: 1m 11s   remaining: 37s
659:    learn: 0.0004824        total: 1m 11s   remaining: 36.9s
660:    learn: 0.0004813        total: 1m 11s   remaining: 36.8s
661:    learn: 0.0004813        total: 1m 11s   remaining: 36.7s
662:    learn: 0.0004813        total: 1m 11s   remaining: 36.6s
663:    learn: 0.0004813        total: 1m 12s   remaining: 36.5s
664:    learn: 0.0004807        total: 1m 12s   remaining: 36.4s
665:    learn: 0.0004805        total: 1m 12s   remaining: 36.2s
666:    learn: 0.0004805        total: 1m 12s   remaining: 36.1s
667:    learn: 0.0004804        total: 1m 12s   remaining: 36s
668:    learn: 0.0004803        total: 1m 12s   remaining: 35.9s
```

```
669:    learn: 0.0004803        total: 1m 12s   remaining: 35.8s
670:    learn: 0.0004792        total: 1m 12s   remaining: 35.7s
671:    learn: 0.0004790        total: 1m 12s   remaining: 35.6s
672:    learn: 0.0004790        total: 1m 13s   remaining: 35.5s
673:    learn: 0.0004790        total: 1m 13s   remaining: 35.4s
674:    learn: 0.0004790        total: 1m 13s   remaining: 35.3s
675:    learn: 0.0004788        total: 1m 13s   remaining: 35.1s
676:    learn: 0.0004788        total: 1m 13s   remaining: 35s
677:    learn: 0.0004788        total: 1m 13s   remaining: 34.9s
678:    learn: 0.0004785        total: 1m 13s   remaining: 34.8s
679:    learn: 0.0004785        total: 1m 13s   remaining: 34.7s
680:    learn: 0.0004785        total: 1m 13s   remaining: 34.6s
681:    learn: 0.0004785        total: 1m 13s   remaining: 34.5s
682:    learn: 0.0004785        total: 1m 14s   remaining: 34.4s
683:    learn: 0.0004784        total: 1m 14s   remaining: 34.3s
684:    learn: 0.0004784        total: 1m 14s   remaining: 34.2s
685:    learn: 0.0004784        total: 1m 14s   remaining: 34.1s
686:    learn: 0.0004784        total: 1m 14s   remaining: 33.9s
687:    learn: 0.0004783        total: 1m 14s   remaining: 33.8s
688:    learn: 0.0004781        total: 1m 14s   remaining: 33.7s
689:    learn: 0.0004781        total: 1m 14s   remaining: 33.6s
690:    learn: 0.0004760        total: 1m 14s   remaining: 33.5s
691:    learn: 0.0004760        total: 1m 15s   remaining: 33.4s
692:    learn: 0.0004760        total: 1m 15s   remaining: 33.3s
693:    learn: 0.0004760        total: 1m 15s   remaining: 33.2s
694:    learn: 0.0004760        total: 1m 15s   remaining: 33.1s
695:    learn: 0.0004752        total: 1m 15s   remaining: 33s
696:    learn: 0.0004742        total: 1m 15s   remaining: 32.8s
697:    learn: 0.0004741        total: 1m 15s   remaining: 32.7s
698:    learn: 0.0004740        total: 1m 15s   remaining: 32.6s
699:    learn: 0.0004738        total: 1m 15s   remaining: 32.5s
700:    learn: 0.0004738        total: 1m 15s   remaining: 32.4s
701:    learn: 0.0004738        total: 1m 16s   remaining: 32.3s
702:    learn: 0.0004738        total: 1m 16s   remaining: 32.2s
703:    learn: 0.0004738        total: 1m 16s   remaining: 32.1s
704:    learn: 0.0004738        total: 1m 16s   remaining: 32s
705:    learn: 0.0004723        total: 1m 16s   remaining: 31.8s
706:    learn: 0.0004722        total: 1m 16s   remaining: 31.7s
707:    learn: 0.0004716        total: 1m 16s   remaining: 31.6s
708:    learn: 0.0004716        total: 1m 16s   remaining: 31.5s
709:    learn: 0.0004716        total: 1m 16s   remaining: 31.4s
710:    learn: 0.0004716        total: 1m 17s   remaining: 31.3s
711:    learn: 0.0004715        total: 1m 17s   remaining: 31.2s
712:    learn: 0.0004714        total: 1m 17s   remaining: 31.1s
713:    learn: 0.0004714        total: 1m 17s   remaining: 31s
714:    learn: 0.0004714        total: 1m 17s   remaining: 30.9s
715:    learn: 0.0004713        total: 1m 17s   remaining: 30.8s
716:    learn: 0.0004713        total: 1m 17s   remaining: 30.7s
717:    learn: 0.0004712        total: 1m 17s   remaining: 30.6s
718:    learn: 0.0004712        total: 1m 17s   remaining: 30.4s
719:    learn: 0.0004711        total: 1m 18s   remaining: 30.3s
720:    learn: 0.0004711        total: 1m 18s   remaining: 30.2s
721:    learn: 0.0004711        total: 1m 18s   remaining: 30.1s
722:    learn: 0.0004710        total: 1m 18s   remaining: 30s
723:    learn: 0.0004710        total: 1m 18s   remaining: 29.9s
724:    learn: 0.0004710        total: 1m 18s   remaining: 29.8s
725:    learn: 0.0004710        total: 1m 18s   remaining: 29.7s
726:    learn: 0.0004708        total: 1m 18s   remaining: 29.6s
727:    learn: 0.0004707        total: 1m 18s   remaining: 29.5s
728:    learn: 0.0004707        total: 1m 18s   remaining: 29.4s
729:    learn: 0.0004707        total: 1m 19s   remaining: 29.3s
730:    learn: 0.0004707        total: 1m 19s   remaining: 29.1s
731:    learn: 0.0004707        total: 1m 19s   remaining: 29s
```

```
732:    learn: 0.0004705        total: 1m 19s   remaining: 28.9s
733:    learn: 0.0004705        total: 1m 19s   remaining: 28.8s
734:    learn: 0.0004705        total: 1m 19s   remaining: 28.7s
735:    learn: 0.0004705        total: 1m 19s   remaining: 28.6s
736:    learn: 0.0004705        total: 1m 19s   remaining: 28.5s
737:    learn: 0.0004705        total: 1m 19s   remaining: 28.4s
738:    learn: 0.0004705        total: 1m 20s   remaining: 28.3s
739:    learn: 0.0004705        total: 1m 20s   remaining: 28.2s
740:    learn: 0.0004698        total: 1m 20s   remaining: 28.1s
741:    learn: 0.0004698        total: 1m 20s   remaining: 28s
742:    learn: 0.0004696        total: 1m 20s   remaining: 27.9s
743:    learn: 0.0004696        total: 1m 20s   remaining: 27.7s
744:    learn: 0.0004696        total: 1m 20s   remaining: 27.6s
745:    learn: 0.0004696        total: 1m 20s   remaining: 27.5s
746:    learn: 0.0004696        total: 1m 20s   remaining: 27.4s
747:    learn: 0.0004696        total: 1m 21s   remaining: 27.3s
748:    learn: 0.0004696        total: 1m 21s   remaining: 27.2s
749:    learn: 0.0004693        total: 1m 21s   remaining: 27.1s
750:    learn: 0.0004693        total: 1m 21s   remaining: 27s
751:    learn: 0.0004692        total: 1m 21s   remaining: 26.9s
752:    learn: 0.0004692        total: 1m 21s   remaining: 26.8s
753:    learn: 0.0004681        total: 1m 21s   remaining: 26.7s
754:    learn: 0.0004681        total: 1m 21s   remaining: 26.6s
755:    learn: 0.0004661        total: 1m 21s   remaining: 26.4s
756:    learn: 0.0004661        total: 1m 22s   remaining: 26.3s
757:    learn: 0.0004661        total: 1m 22s   remaining: 26.2s
758:    learn: 0.0004660        total: 1m 22s   remaining: 26.1s
759:    learn: 0.0004659        total: 1m 22s   remaining: 26s
760:    learn: 0.0004659        total: 1m 22s   remaining: 25.9s
761:    learn: 0.0004659        total: 1m 22s   remaining: 25.8s
762:    learn: 0.0004659        total: 1m 22s   remaining: 25.7s
763:    learn: 0.0004659        total: 1m 22s   remaining: 25.6s
764:    learn: 0.0004659        total: 1m 22s   remaining: 25.5s
765:    learn: 0.0004659        total: 1m 23s   remaining: 25.4s
766:    learn: 0.0004651        total: 1m 23s   remaining: 25.3s
767:    learn: 0.0004651        total: 1m 23s   remaining: 25.1s
768:    learn: 0.0004650        total: 1m 23s   remaining: 25s
769:    learn: 0.0004650        total: 1m 23s   remaining: 24.9s
770:    learn: 0.0004648        total: 1m 23s   remaining: 24.8s
771:    learn: 0.0004648        total: 1m 23s   remaining: 24.7s
772:    learn: 0.0004648        total: 1m 23s   remaining: 24.6s
773:    learn: 0.0004648        total: 1m 23s   remaining: 24.5s
774:    learn: 0.0004648        total: 1m 24s   remaining: 24.4s
775:    learn: 0.0004648        total: 1m 24s   remaining: 24.3s
776:    learn: 0.0004647        total: 1m 24s   remaining: 24.2s
777:    learn: 0.0004647        total: 1m 24s   remaining: 24.1s
778:    learn: 0.0004647        total: 1m 24s   remaining: 24s
779:    learn: 0.0004647        total: 1m 24s   remaining: 23.9s
780:    learn: 0.0004647        total: 1m 24s   remaining: 23.7s
781:    learn: 0.0004646        total: 1m 24s   remaining: 23.6s
782:    learn: 0.0004646        total: 1m 24s   remaining: 23.5s
783:    learn: 0.0004646        total: 1m 24s   remaining: 23.4s
784:    learn: 0.0004646        total: 1m 25s   remaining: 23.3s
785:    learn: 0.0004646        total: 1m 25s   remaining: 23.2s
786:    learn: 0.0004646        total: 1m 25s   remaining: 23.1s
787:    learn: 0.0004636        total: 1m 25s   remaining: 23s
788:    learn: 0.0004635        total: 1m 25s   remaining: 22.9s
789:    learn: 0.0004621        total: 1m 25s   remaining: 22.8s
790:    learn: 0.0004621        total: 1m 25s   remaining: 22.7s
791:    learn: 0.0004620        total: 1m 25s   remaining: 22.6s
792:    learn: 0.0004610        total: 1m 26s   remaining: 22.5s
793:    learn: 0.0004609        total: 1m 26s   remaining: 22.4s
794:    learn: 0.0004595        total: 1m 26s   remaining: 22.3s
```

```
795:    learn: 0.0004594        total: 1m 26s   remaining: 22.1s
796:    learn: 0.0004594        total: 1m 26s   remaining: 22s
797:    learn: 0.0004593        total: 1m 26s   remaining: 21.9s
798:    learn: 0.0004593        total: 1m 26s   remaining: 21.8s
799:    learn: 0.0004593        total: 1m 26s   remaining: 21.7s
800:    learn: 0.0004593        total: 1m 27s   remaining: 21.6s
801:    learn: 0.0004593        total: 1m 27s   remaining: 21.5s
802:    learn: 0.0004593        total: 1m 27s   remaining: 21.4s
803:    learn: 0.0004592        total: 1m 27s   remaining: 21.3s
804:    learn: 0.0004591        total: 1m 27s   remaining: 21.2s
805:    learn: 0.0004590        total: 1m 27s   remaining: 21.1s
806:    learn: 0.0004585        total: 1m 27s   remaining: 21s
807:    learn: 0.0004585        total: 1m 27s   remaining: 20.9s
808:    learn: 0.0004585        total: 1m 27s   remaining: 20.8s
809:    learn: 0.0004585        total: 1m 28s   remaining: 20.7s
810:    learn: 0.0004584        total: 1m 28s   remaining: 20.6s
811:    learn: 0.0004584        total: 1m 28s   remaining: 20.4s
812:    learn: 0.0004584        total: 1m 28s   remaining: 20.3s
813:    learn: 0.0004584        total: 1m 28s   remaining: 20.2s
814:    learn: 0.0004583        total: 1m 28s   remaining: 20.1s
815:    learn: 0.0004583        total: 1m 28s   remaining: 20s
816:    learn: 0.0004566        total: 1m 28s   remaining: 19.9s
817:    learn: 0.0004564        total: 1m 28s   remaining: 19.8s
818:    learn: 0.0004564        total: 1m 29s   remaining: 19.7s
819:    learn: 0.0004564        total: 1m 29s   remaining: 19.6s
820:    learn: 0.0004551        total: 1m 29s   remaining: 19.5s
821:    learn: 0.0004549        total: 1m 29s   remaining: 19.3s
822:    learn: 0.0004549        total: 1m 29s   remaining: 19.2s
823:    learn: 0.0004549        total: 1m 29s   remaining: 19.1s
824:    learn: 0.0004549        total: 1m 29s   remaining: 19s
825:    learn: 0.0004549        total: 1m 29s   remaining: 18.9s
826:    learn: 0.0004543        total: 1m 29s   remaining: 18.8s
827:    learn: 0.0004537        total: 1m 30s   remaining: 18.7s
828:    learn: 0.0004537        total: 1m 30s   remaining: 18.6s
829:    learn: 0.0004535        total: 1m 30s   remaining: 18.5s
830:    learn: 0.0004535        total: 1m 30s   remaining: 18.4s
831:    learn: 0.0004535        total: 1m 30s   remaining: 18.3s
832:    learn: 0.0004534        total: 1m 30s   remaining: 18.2s
833:    learn: 0.0004534        total: 1m 30s   remaining: 18s
834:    learn: 0.0004534        total: 1m 30s   remaining: 17.9s
835:    learn: 0.0004534        total: 1m 30s   remaining: 17.8s
836:    learn: 0.0004533        total: 1m 30s   remaining: 17.7s
837:    learn: 0.0004533        total: 1m 31s   remaining: 17.6s
838:    learn: 0.0004533        total: 1m 31s   remaining: 17.5s
839:    learn: 0.0004516        total: 1m 31s   remaining: 17.4s
840:    learn: 0.0004510        total: 1m 31s   remaining: 17.3s
841:    learn: 0.0004510        total: 1m 31s   remaining: 17.2s
842:    learn: 0.0004510        total: 1m 31s   remaining: 17.1s
843:    learn: 0.0004510        total: 1m 31s   remaining: 17s
844:    learn: 0.0004505        total: 1m 31s   remaining: 16.8s
845:    learn: 0.0004505        total: 1m 31s   remaining: 16.7s
846:    learn: 0.0004505        total: 1m 32s   remaining: 16.6s
847:    learn: 0.0004504        total: 1m 32s   remaining: 16.5s
848:    learn: 0.0004504        total: 1m 32s   remaining: 16.4s
849:    learn: 0.0004503        total: 1m 32s   remaining: 16.3s
850:    learn: 0.0004503        total: 1m 32s   remaining: 16.2s
851:    learn: 0.0004503        total: 1m 32s   remaining: 16.1s
852:    learn: 0.0004503        total: 1m 32s   remaining: 16s
853:    learn: 0.0004503        total: 1m 32s   remaining: 15.9s
854:    learn: 0.0004503        total: 1m 32s   remaining: 15.8s
855:    learn: 0.0004503        total: 1m 33s   remaining: 15.6s
856:    learn: 0.0004503        total: 1m 33s   remaining: 15.5s
857:    learn: 0.0004502        total: 1m 33s   remaining: 15.4s
```

```
858:    learn: 0.0004502        total: 1m 33s   remaining: 15.3s
859:    learn: 0.0004502        total: 1m 33s   remaining: 15.2s
860:    learn: 0.0004502        total: 1m 33s   remaining: 15.1s
861:    learn: 0.0004502        total: 1m 33s   remaining: 15s
862:    learn: 0.0004501        total: 1m 33s   remaining: 14.9s
863:    learn: 0.0004500        total: 1m 33s   remaining: 14.8s
864:    learn: 0.0004500        total: 1m 33s   remaining: 14.7s
865:    learn: 0.0004499        total: 1m 34s   remaining: 14.6s
866:    learn: 0.0004499        total: 1m 34s   remaining: 14.4s
867:    learn: 0.0004498        total: 1m 34s   remaining: 14.3s
868:    learn: 0.0004497        total: 1m 34s   remaining: 14.2s
869:    learn: 0.0004497        total: 1m 34s   remaining: 14.1s
870:    learn: 0.0004497        total: 1m 34s   remaining: 14s
871:    learn: 0.0004497        total: 1m 34s   remaining: 13.9s
872:    learn: 0.0004493        total: 1m 34s   remaining: 13.8s
873:    learn: 0.0004493        total: 1m 34s   remaining: 13.7s
874:    learn: 0.0004493        total: 1m 34s   remaining: 13.6s
875:    learn: 0.0004492        total: 1m 35s   remaining: 13.5s
876:    learn: 0.0004491        total: 1m 35s   remaining: 13.3s
877:    learn: 0.0004490        total: 1m 35s   remaining: 13.2s
878:    learn: 0.0004489        total: 1m 35s   remaining: 13.1s
879:    learn: 0.0004477        total: 1m 35s   remaining: 13s
880:    learn: 0.0004477        total: 1m 35s   remaining: 12.9s
881:    learn: 0.0004477        total: 1m 35s   remaining: 12.8s
882:    learn: 0.0004476        total: 1m 35s   remaining: 12.7s
883:    learn: 0.0004476        total: 1m 35s   remaining: 12.6s
884:    learn: 0.0004476        total: 1m 36s   remaining: 12.5s
885:    learn: 0.0004475        total: 1m 36s   remaining: 12.4s
886:    learn: 0.0004475        total: 1m 36s   remaining: 12.3s
887:    learn: 0.0004474        total: 1m 36s   remaining: 12.2s
888:    learn: 0.0004474        total: 1m 36s   remaining: 12s
889:    learn: 0.0004473        total: 1m 36s   remaining: 11.9s
890:    learn: 0.0004472        total: 1m 36s   remaining: 11.8s
891:    learn: 0.0004464        total: 1m 36s   remaining: 11.7s
892:    learn: 0.0004463        total: 1m 36s   remaining: 11.6s
893:    learn: 0.0004463        total: 1m 37s   remaining: 11.5s
894:    learn: 0.0004463        total: 1m 37s   remaining: 11.4s
895:    learn: 0.0004463        total: 1m 37s   remaining: 11.3s
896:    learn: 0.0004454        total: 1m 37s   remaining: 11.2s
897:    learn: 0.0004454        total: 1m 37s   remaining: 11.1s
898:    learn: 0.0004451        total: 1m 37s   remaining: 11s
899:    learn: 0.0004451        total: 1m 37s   remaining: 10.9s
900:    learn: 0.0004450        total: 1m 37s   remaining: 10.7s
901:    learn: 0.0004450        total: 1m 37s   remaining: 10.6s
902:    learn: 0.0004449        total: 1m 38s   remaining: 10.5s
903:    learn: 0.0004449        total: 1m 38s   remaining: 10.4s
904:    learn: 0.0004449        total: 1m 38s   remaining: 10.3s
905:    learn: 0.0004449        total: 1m 38s   remaining: 10.2s
906:    learn: 0.0004449        total: 1m 38s   remaining: 10.1s
907:    learn: 0.0004449        total: 1m 38s   remaining: 9.98s
908:    learn: 0.0004448        total: 1m 38s   remaining: 9.87s
909:    learn: 0.0004448        total: 1m 38s   remaining: 9.77s
910:    learn: 0.0004448        total: 1m 38s   remaining: 9.66s
911:    learn: 0.0004448        total: 1m 38s   remaining: 9.55s
912:    learn: 0.0004448        total: 1m 39s   remaining: 9.44s
913:    learn: 0.0004448        total: 1m 39s   remaining: 9.33s
914:    learn: 0.0004448        total: 1m 39s   remaining: 9.22s
915:    learn: 0.0004448        total: 1m 39s   remaining: 9.11s
916:    learn: 0.0004435        total: 1m 39s   remaining: 9s
917:    learn: 0.0004435        total: 1m 39s   remaining: 8.89s
918:    learn: 0.0004435        total: 1m 39s   remaining: 8.79s
919:    learn: 0.0004435        total: 1m 39s   remaining: 8.68s
920:    learn: 0.0004435        total: 1m 39s   remaining: 8.57s
```

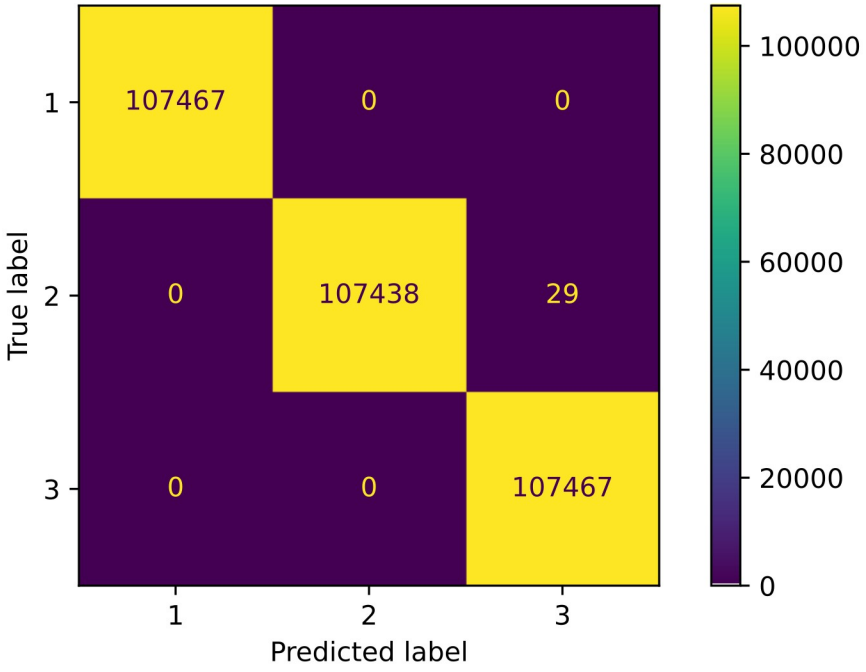
```
921:    learn: 0.0004435        total: 1m 40s   remaining: 8.46s
922:    learn: 0.0004435        total: 1m 40s   remaining: 8.35s
923:    learn: 0.0004434        total: 1m 40s   remaining: 8.24s
924:    learn: 0.0004434        total: 1m 40s   remaining: 8.13s
925:    learn: 0.0004407        total: 1m 40s   remaining: 8.03s
926:    learn: 0.0004401        total: 1m 40s   remaining: 7.92s
927:    learn: 0.0004401        total: 1m 40s   remaining: 7.81s
928:    learn: 0.0004401        total: 1m 40s   remaining: 7.7s
929:    learn: 0.0004401        total: 1m 40s   remaining: 7.59s
930:    learn: 0.0004399        total: 1m 40s   remaining: 7.48s
931:    learn: 0.0004398        total: 1m 41s   remaining: 7.37s
932:    learn: 0.0004398        total: 1m 41s   remaining: 7.26s
933:    learn: 0.0004398        total: 1m 41s   remaining: 7.16s
934:    learn: 0.0004397        total: 1m 41s   remaining: 7.05s
935:    learn: 0.0004397        total: 1m 41s   remaining: 6.94s
936:    learn: 0.0004397        total: 1m 41s   remaining: 6.83s
937:    learn: 0.0004397        total: 1m 41s   remaining: 6.72s
938:    learn: 0.0004392        total: 1m 41s   remaining: 6.61s
939:    learn: 0.0004392        total: 1m 41s   remaining: 6.5s
940:    learn: 0.0004392        total: 1m 42s   remaining: 6.4s
941:    learn: 0.0004392        total: 1m 42s   remaining: 6.29s
942:    learn: 0.0004392        total: 1m 42s   remaining: 6.18s
943:    learn: 0.0004392        total: 1m 42s   remaining: 6.07s
944:    learn: 0.0004392        total: 1m 42s   remaining: 5.96s
945:    learn: 0.0004392        total: 1m 42s   remaining: 5.85s
946:    learn: 0.0004392        total: 1m 42s   remaining: 5.75s
947:    learn: 0.0004391        total: 1m 42s   remaining: 5.64s
948:    learn: 0.0004390        total: 1m 42s   remaining: 5.53s
949:    learn: 0.0004390        total: 1m 42s   remaining: 5.42s
950:    learn: 0.0004390        total: 1m 43s   remaining: 5.31s
951:    learn: 0.0004390        total: 1m 43s   remaining: 5.2s
952:    learn: 0.0004390        total: 1m 43s   remaining: 5.09s
953:    learn: 0.0004389        total: 1m 43s   remaining: 4.99s
954:    learn: 0.0004389        total: 1m 43s   remaining: 4.88s
955:    learn: 0.0004388        total: 1m 43s   remaining: 4.77s
956:    learn: 0.0004387        total: 1m 43s   remaining: 4.66s
957:    learn: 0.0004387        total: 1m 43s   remaining: 4.55s
958:    learn: 0.0004386        total: 1m 43s   remaining: 4.44s
959:    learn: 0.0004386        total: 1m 44s   remaining: 4.33s
960:    learn: 0.0004386        total: 1m 44s   remaining: 4.23s
961:    learn: 0.0004386        total: 1m 44s   remaining: 4.12s
962:    learn: 0.0004386        total: 1m 44s   remaining: 4.01s
963:    learn: 0.0004386        total: 1m 44s   remaining: 3.9s
964:    learn: 0.0004386        total: 1m 44s   remaining: 3.79s
965:    learn: 0.0004386        total: 1m 44s   remaining: 3.68s
966:    learn: 0.0004386        total: 1m 44s   remaining: 3.58s
967:    learn: 0.0004386        total: 1m 44s   remaining: 3.47s
968:    learn: 0.0004386        total: 1m 45s   remaining: 3.36s
969:    learn: 0.0004386        total: 1m 45s   remaining: 3.25s
970:    learn: 0.0004381        total: 1m 45s   remaining: 3.14s
971:    learn: 0.0004381        total: 1m 45s   remaining: 3.03s
972:    learn: 0.0004381        total: 1m 45s   remaining: 2.92s
973:    learn: 0.0004381        total: 1m 45s   remaining: 2.82s
974:    learn: 0.0004381        total: 1m 45s   remaining: 2.71s
975:    learn: 0.0004377        total: 1m 45s   remaining: 2.6s
976:    learn: 0.0004377        total: 1m 45s   remaining: 2.49s
977:    learn: 0.0004377        total: 1m 45s   remaining: 2.38s
978:    learn: 0.0004377        total: 1m 46s   remaining: 2.27s
979:    learn: 0.0004375        total: 1m 46s   remaining: 2.17s
980:    learn: 0.0004374        total: 1m 46s   remaining: 2.06s
981:    learn: 0.0004374        total: 1m 46s   remaining: 1.95s
982:    learn: 0.0004374        total: 1m 46s   remaining: 1.84s
983:    learn: 0.0004373        total: 1m 46s   remaining: 1.73s
```

```
984:    learn: 0.0004373        total: 1m 46s   remaining: 1.62s
985:    learn: 0.0004360        total: 1m 46s   remaining: 1.52s
986:    learn: 0.0004360        total: 1m 46s   remaining: 1.41s
987:    learn: 0.0004356        total: 1m 47s   remaining: 1.3s
988:    learn: 0.0004356        total: 1m 47s   remaining: 1.19s
989:    learn: 0.0004356        total: 1m 47s   remaining: 1.08s
990:    learn: 0.0004356        total: 1m 47s   remaining: 975ms
991:    learn: 0.0004356        total: 1m 47s   remaining: 867ms
992:    learn: 0.0004356        total: 1m 47s   remaining: 758ms
993:    learn: 0.0004356        total: 1m 47s   remaining: 650ms
994:    learn: 0.0004355        total: 1m 47s   remaining: 542ms
995:    learn: 0.0004355        total: 1m 47s   remaining: 433ms
996:    learn: 0.0004355        total: 1m 47s   remaining: 325ms
```

In [151…

```
display_results(y_train_os, y_pred_train_os_cb, clf_os_cb)
```

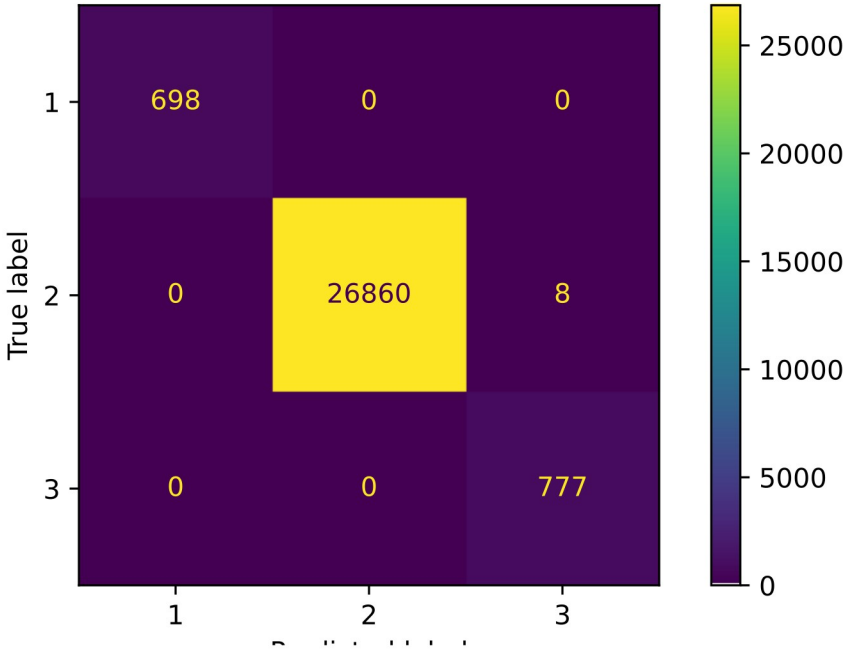
```
              precision    recall  f1-score   support

           1       1.00      1.00      1.00    107467
           2       1.00      1.00      1.00    107467
           3       1.00      1.00      1.00    107467

    accuracy                           1.00    322401
   macro avg       1.00      1.00      1.00    322401
weighted avg       1.00      1.00      1.00    322401
```



In [152…

```
display_results(y_test, y_pred_test_os_cb, clf_os_cb)
```

```
              precision    recall  f1-score   support

           1       1.00      1.00      1.00       698
           2       1.00      1.00      1.00     26868
           3       0.99      1.00      0.99       777

    accuracy                           1.00     28343
   macro avg       1.00      1.00      1.00     28343
weighted avg       1.00      1.00      1.00     28343
```



5.4.6 LGBM

In [153…
```
from lightgbm import LGBMClassifier
```

In [154…
```
y_pred_train_os_lb, y_pred_test_os_lb, clf_os_lb =clf_model(Xtrain=X_train_
          Xtest=X_test1,
          ytrain=y_train_os,
          ytest=y_test,
          classifier=LGBMClassifier(random_state=42))
```

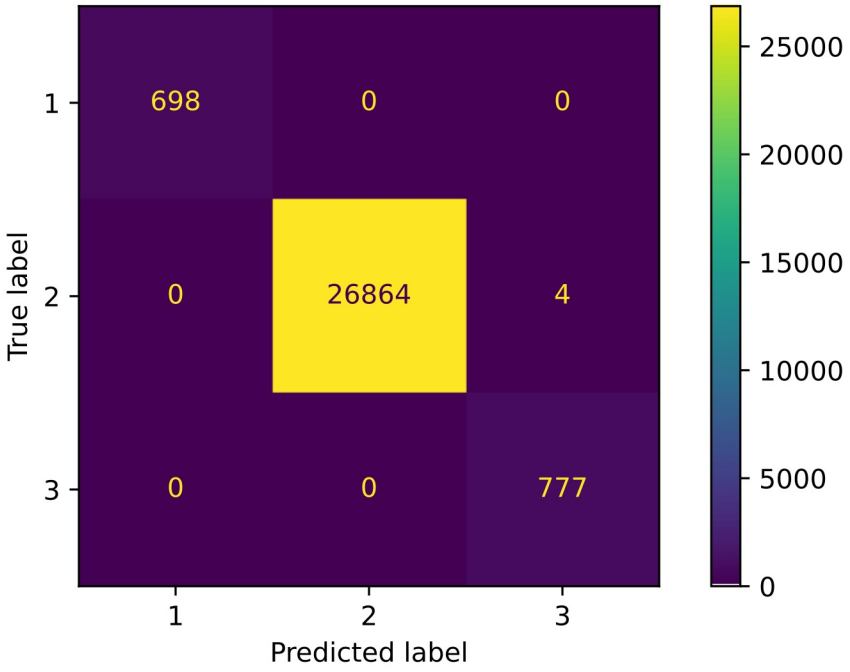
In [155…
```
display_results(y_train_os, y_pred_train_os_lb, clf_os_lb)
```

```
              precision    recall  f1-score   support

           1       1.00      1.00      1.00    107467
           2       1.00      1.00      1.00    107467
           3       1.00      1.00      1.00    107467

    accuracy                           1.00    322401
   macro avg       1.00      1.00      1.00    322401
weighted avg       1.00      1.00      1.00    322401
```

In [156…

```
display_results(y_test, y_pred_test_os_lb, clf_os_lb)
```

```
              precision    recall  f1-score   support

           1       1.00      1.00      1.00       698
           2       1.00      1.00      1.00     26868
           3       0.99      1.00      1.00       777

    accuracy                           1.00     28343
   macro avg       1.00      1.00      1.00     28343
weighted avg       1.00      1.00      1.00     28343
```



5.4.7 KNN

In [157…

```
from sklearn.neighbors import KNeighborsClassifier
```

In [158…

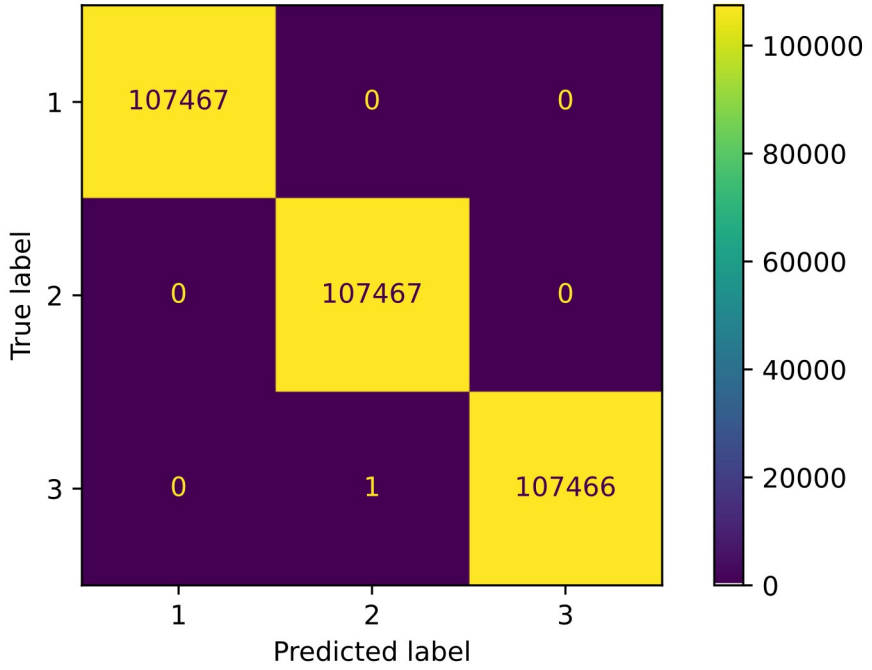
```
y_pred_train_os_kn, y_pred_test_os_kn, clf_os_kn =clf_model(Xtrain=X_train_
          Xtest=X_test1,
          ytrain=y_train_os,
          ytest=y_test,
          classifier=KNeighborsClassifier(n_neighbors=1))
```

In [159…

```
display_results(y_train_os, y_pred_train_os_kn, clf_os_kn)
```

```
              precision    recall  f1-score   support

           1       1.00      1.00      1.00    107467
           2       1.00      1.00      1.00    107467
           3       1.00      1.00      1.00    107467
```

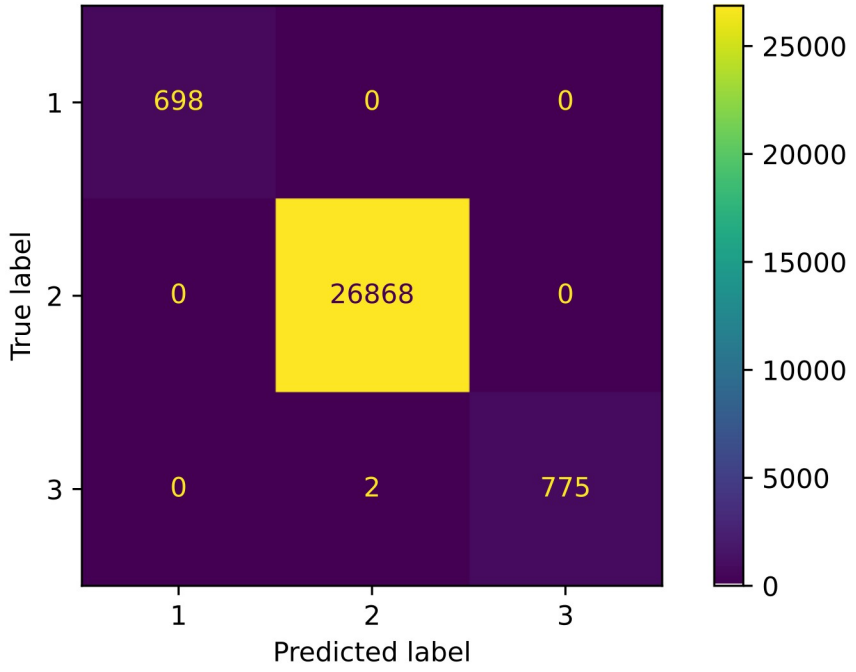
accuracy			1.00	322401
macro avg	1.00	1.00	1.00	322401
weighted avg	1.00	1.00	1.00	322401



In [160…

```
display_results(y_test, y_pred_test_os_kn, clf_os_kn)
```

	precision	recall	f1-score	support
1	1.00	1.00	1.00	698
2	1.00	1.00	1.00	26868
3	1.00	1.00	1.00	777
accuracy			1.00	28343
macro avg	1.00	1.00	1.00	28343
weighted avg	1.00	1.00	1.00	28343



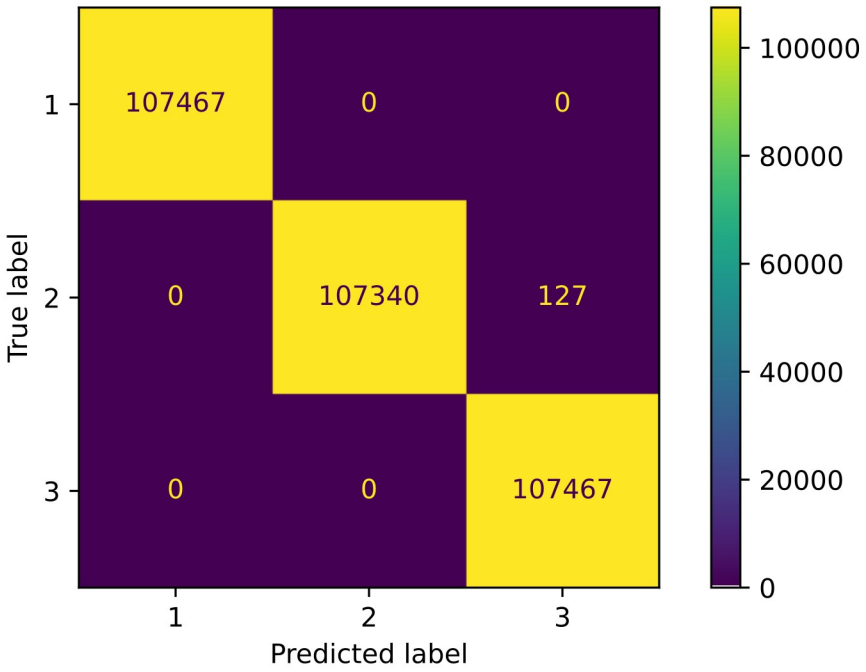
5.4.8 SVM

In [161…
```
from sklearn.svm import SVC
```

In [162…
```
y_pred_train_os_sv, y_pred_test_os_sv, clf_os_sv =clf_model(Xtrain=X_train_
          Xtest=X_test1,
          ytrain=y_train_os,
          ytest=y_test,
          classifier=SVC())
```

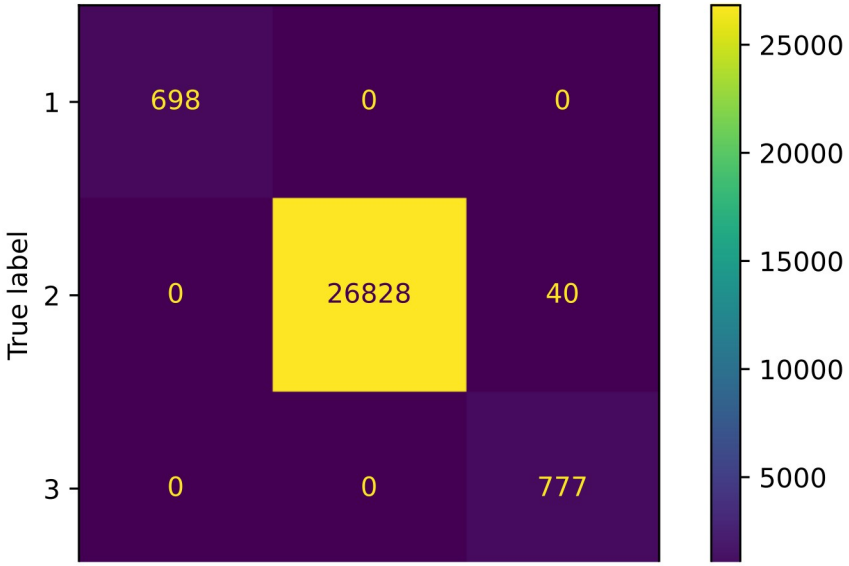
In [163…
```
display_results(y_train_os, y_pred_train_os_sv, clf_os_sv)
```

```
              precision    recall  f1-score   support

           1       1.00      1.00      1.00    107467
           2       1.00      1.00      1.00    107467
           3       1.00      1.00      1.00    107467

    accuracy                           1.00    322401
   macro avg       1.00      1.00      1.00    322401
weighted avg       1.00      1.00      1.00    322401
```



In [164…
```
display_results(y_test, y_pred_test_os_sv, clf_os_sv)
```

```
              precision    recall  f1-score   support

           1       1.00      1.00      1.00       698
           2       1.00      1.00      1.00     26868
           3       0.95      1.00      0.97       777

    accuracy                           1.00     28343
   macro avg       0.98      1.00      0.99     28343
weighted avg       1.00      1.00      1.00     28343
```



5.4.9 Gaussian Naive Bayes

In [165…
```
from sklearn.naive_bayes import GaussianNB
```

In [166…
```
y_pred_train_os_gnb, y_pred_test_os_gnb, clf_os_gnb =clf_model(Xtrain=X_tra
          Xtest=X_test1,
          ytrain=y_train_os,
          ytest=y_test,
          classifier=GaussianNB())
```

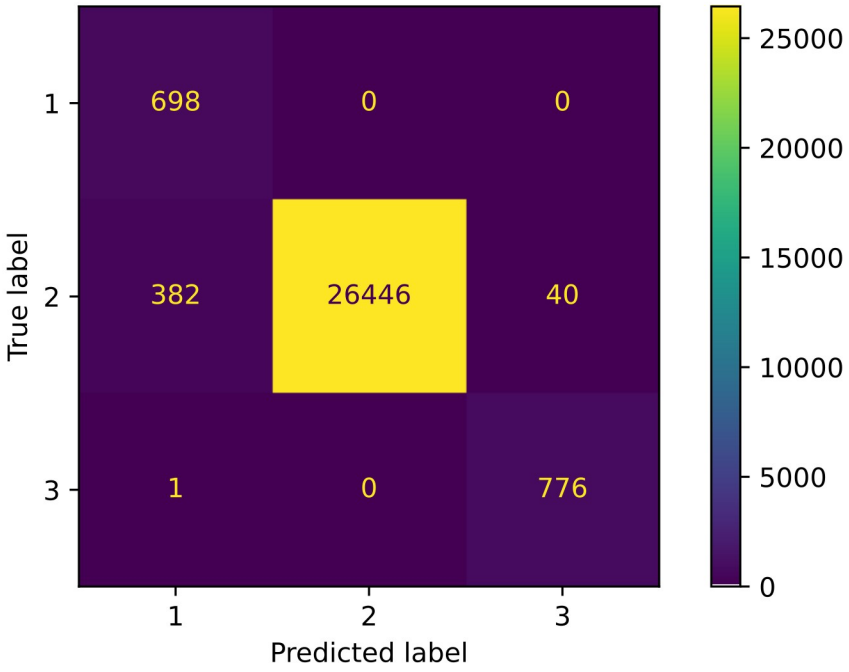
In [167…
```
display_results(y_train_os, y_pred_train_os_gnb, clf_os_gnb)
```

```
              precision    recall  f1-score   support

           1       0.98      1.00      0.99    107467
           2       1.00      0.99      0.99    107467
           3       1.00      0.99      1.00    107467

    accuracy                           0.99    322401
   macro avg       0.99      0.99      0.99    322401
weighted avg       0.99      0.99      0.99    322401
```

In [168…

```
display_results(y_test, y_pred_test_os_gnb, clf_os_gnb)
```

```
              precision    recall  f1-score   support

           1       0.65      1.00      0.78       698
           2       1.00      0.98      0.99     26868
           3       0.95      1.00      0.97       777

    accuracy                           0.99     28343
   macro avg       0.87      0.99      0.92     28343
weighted avg       0.99      0.99      0.99     28343
```



5.4.10 Logistic Regression

In [169…

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

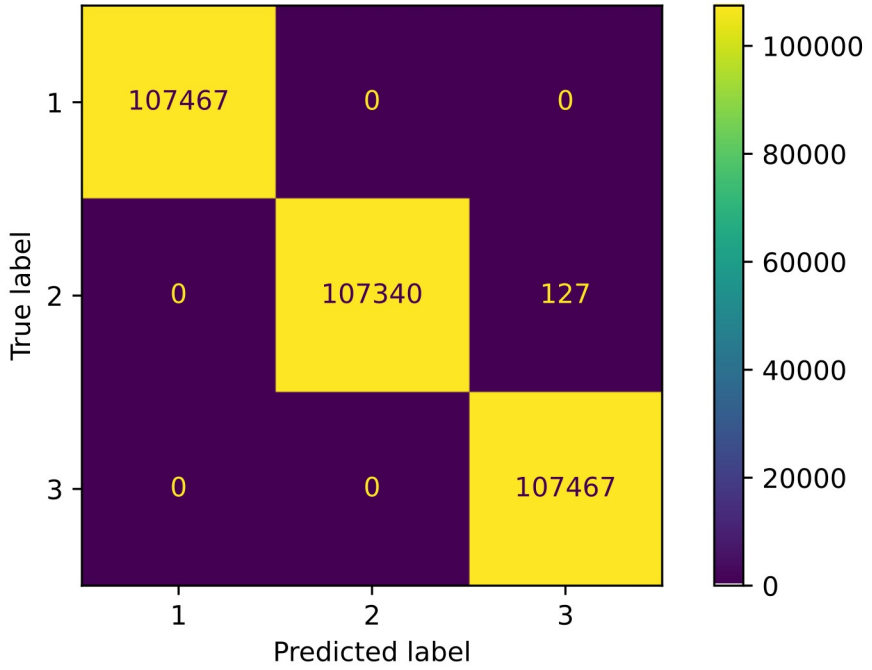
In [170…

```
y_pred_train_os_log, y_pred_test_os_log, clf_os_log =clf_model(Xtrain=X_tra
          Xtest=X_test1,
          ytrain=y_train_os,
          ytest=y_test,
          classifier=LogisticRegression())
```

In [171…

```
display_results(y_train_os, y_pred_train_os_log, clf_os_log)
```

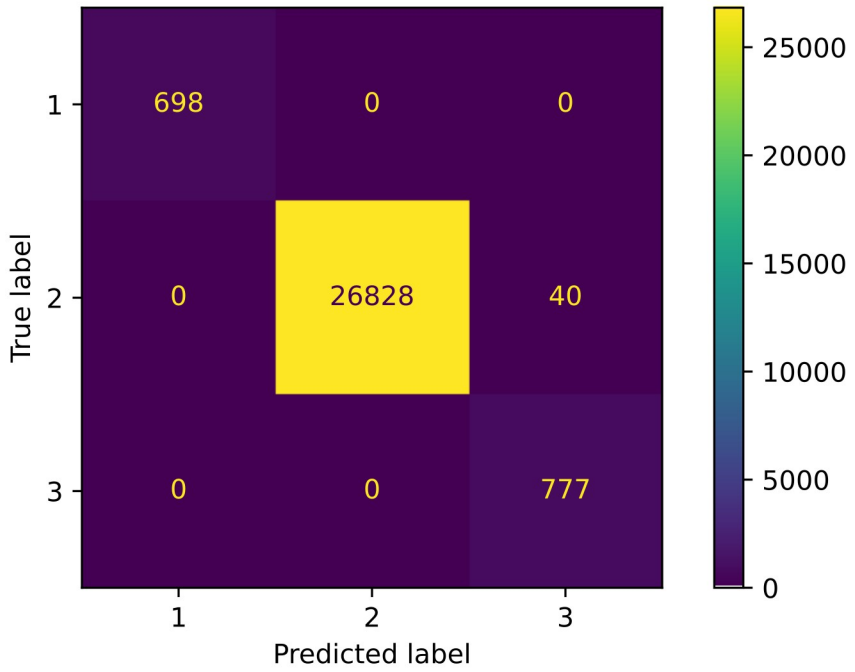
```
              precision    recall  f1-score   support

           1       1.00      1.00      1.00    107467
           2       1.00      1.00      1.00    107467
           3       1.00      1.00      1.00    107467

    accuracy                           1.00    322401
   macro avg       1.00      1.00      1.00    322401
weighted avg       1.00      1.00      1.00    322401
```



In [172…

```
display_results(y_test, y_pred_test_os_log, clf_os_log)
```

	precision	recall	f1-score	support
1	1.00	1.00	1.00	698
2	1.00	1.00	1.00	26868
3	0.95	1.00	0.97	777
accuracy			1.00	28343
macro avg	0.98	1.00	0.99	28343
weighted avg	1.00	1.00	1.00	28343



6 Creating sample predictions for chosen model

Chosen model - Random forest

In [183...
```
test_df = pd.read_csv('incident_impact_sample.csv') # opening a prepared da
```

In [184...
```
sel_features
```

Out[184...
```
['urgency', 'number', 'opened_by', 'priority']
```

In [185...
```
# sel_features = ['number', 'priority', 'urgency', 'opened_by']
test_df
```

Out[185...

	urgency	number	opened_by	priority	impact
0	2	29727	24	3	2
1	2	1287	468	3	2
2	2	15348	17	3	2
3	2	22969	305	2	1
4	2	30514	58	2	1
5	2	22377	24	2	1
6	3	33660	131	4	3
7	3	26073	20	4	3
8	3	2285	533	4	3

In [186...
```
X_sample = test_df.iloc[:, :-1]
y_sample = test_df.iloc[:, -1]
```

In [187...
```
sc = MinMaxScaler()
X_sample = pd.DataFrame(scaler.fit_transform(X_sample), columns=X_sample.co
```

In [188...
```
X_sample
```

Out[188...

	urgency	number	opened_by	priority
0	0.0	0.878510	0.013566	0.5
1	0.0	0.000000	0.874031	0.5
2	0.0	0.434343	0.000000	0.5
3	0.0	0.669756	0.558140	0.0
4	0.0	0.902820	0.079457	0.0
5	0.0	0.651469	0.013566	0.0
6	1.0	1.000000	0.220930	1.0
7	1.0	0.765638	0.005814	1.0
8	1.0	0.030828	1.000000	1.0

In [189…
```
y_sample_pred = pd.DataFrame(clf_rf.predict(X_sample), columns=['impact_pre
```

In [190…
```
y_sample_pred
```

Out[190…

	impact_pred
0	2
1	2
2	2
3	1
4	1
5	1
6	3
7	3
8	3

In [191…
```
predictions = pd.concat([test_df,y_sample_pred], axis=1)
```

In [192…
```
predictions.to_csv('impact_preds.csv')
```