In [1]:
```
# 12:25 PM
# 12:35 PM
import os; print(os.path.dirname(os.getcwd()).split('\\')[-1])
```

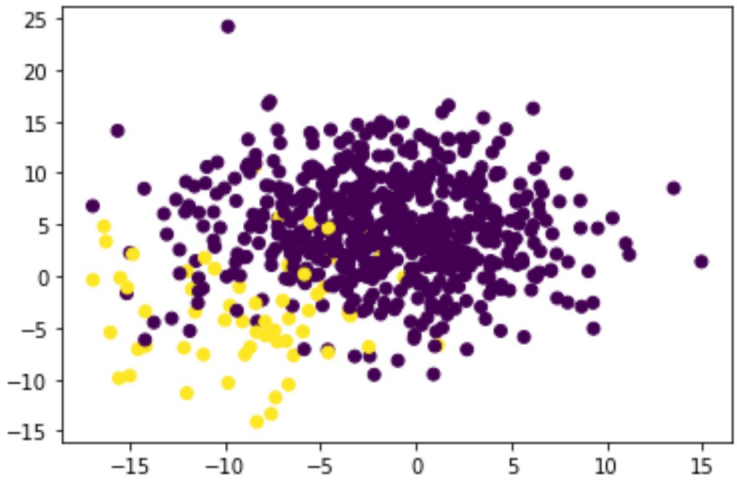
```
01-Ins_Review_Eval_Metrics
```

In [2]:
```
%matplotlib inline
import matplotlib.pyplot as plt
import pandas as pd
from sklearn.datasets import make_blobs
from collections import Counter
```

In [3]:
```
# Generate imbalanced dataset
X, y = make_blobs(n_samples = [600,60], random_state=1, cluster_std=5)

plt.scatter(X[:, 0], X[:, 1], c=y)
```

Out[3]:
```
<matplotlib.collections.PathCollection at 0x12908af3488>
```



Comment

Note - the data is imbalanced, and there is major overlap in the data, confounding classification attempts.

NB - Split before resampling.

In [4]:
```
# Normal train-test split
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, random_state=1)
```

In [5]:
```
Counter(y_train)
```

Out[5]:
```
Counter({0: 451, 1: 44})
```

Comment

Downside of SMOTE is that it doesn't see the overall distribution of the data. Therefore, it can create points that are heavily influenced by outliers, and thus generates some poor data points.

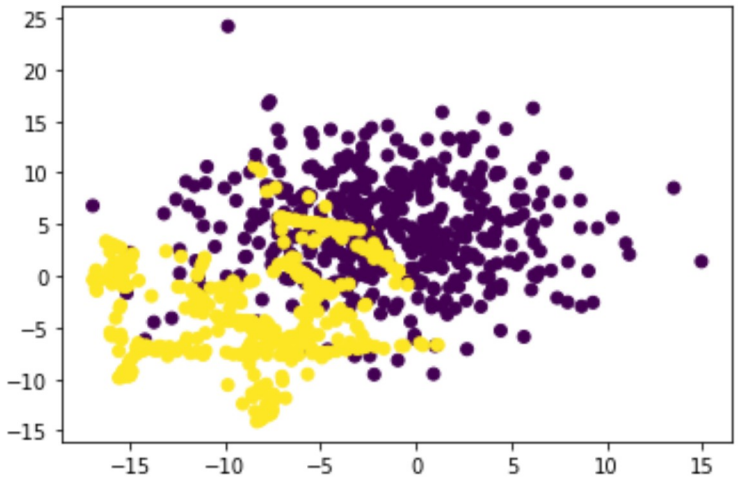
In [6]:
```
# SMOTE oversampling
from imblearn.over_sampling import SMOTE
from collections import Counter

X_resampled, y_resampled = SMOTE(random_state=1, sampling_strategy=1.0).fit_re
Counter(y_resampled)
```

Out[6]:
```
Counter({0: 451, 1: 451})
```

In [7]:
```
plt.scatter(X_resampled[:, 0], X_resampled[:, 1], c=y_resampled)
```

Out[7]:
```
<matplotlib.collections.PathCollection at 0x1290ab423c8>
```



Comment

SMOTEENN

ENN in SMOTEENN stands for edited nearest-neighbor rule. It looks at the labels in the sampled data and removes the ones that are surrounded by the opposite class. This clears out the indistinct edge cases, pruning the data.

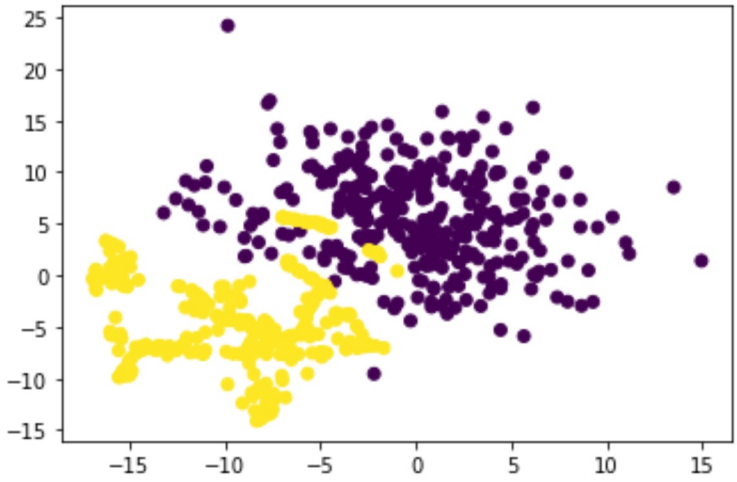
In [8]:
```
#SMOTEENN combination sampling
from imblearn.combine import SMOTEENN

sm = SMOTEENN(random_state=1)
X_resampled, y_resampled = sm.fit_resample(X_train, y_train)
Counter(y_resampled)
```

Out[8]:
```
Counter({0: 330, 1: 361})
```

In [9]:
```
plt.scatter(X_resampled[:, 0], X_resampled[:, 1], c=y_resampled)
```

Out[9]:
```
<matplotlib.collections.PathCollection at 0x1290abc6708>
```



Comment

Less overlap! SMOTEENN differentiated the data and removed outliers in each class.

In [10]:
```
# Logistic regression using random combination sampled data
from sklearn.linear_model import LogisticRegression

model = LogisticRegression(solver='lbfgs', random_state=1)
model.fit(X_resampled, y_resampled)
```

Out[10]:
```
LogisticRegression(random_state=1)
```

In [11]:
```
# Display the confusion matrix
from sklearn.metrics import confusion_matrix

y_pred = model.predict(X_test)
confusion_matrix(y_test, y_pred)
```

Out[11]:
```
array([[126,  23],
       [  1,  15]], dtype=int64)
```

In [12]:
```
# Print the imbalanced classification report
from imblearn.metrics import classification_report_imbalanced

print(classification_report_imbalanced(y_test, y_pred))
```

```
                   pre       rec       spe        f1       geo       iba
sup

          0       0.99      0.85      0.94      0.91      0.89      0.79
149
          1       0.39      0.94      0.85      0.56      0.89      0.80
16

avg / total       0.93      0.85      0.93      0.88      0.89      0.79
165
```

In [ ]:
```

```

In [ ]: