# Measuring Classification Performance

## Cross-validation and accuracy

```
from sklearn.model_selection import cross_val_score
scores = cross_val_score(log_reg, normalised_train_df, y_balanced, cv=5, scoring='f1_macro')
scores
#prints
array([0.55594592, 0.4733312 , 0.55651249, 0.5245098 , 0.58315241])
```

## K-Fold Cross Validation

```
from sklearn.model_selection import KFold
kf = KFold(n_splits=5)
kf.split(normalised_train_df)
f1_scores = []
#run for every split
for train_index, test_index in kf.split(normalised_train_df):
  x_train, x_test = normalised_train_df.iloc[train_index],
                    normalised_train_df.iloc[test_index]
  y_train, y_test = y_balanced[train_index],
                    y_balanced[test_index]
  model = LogisticRegression().fit(x_train, y_train)
  #save result to list
  f1_scores.append(f1_score(y_true=y_test, y_pred=model.predict(x_test),
                  pos_label='2A')*100)
```

## Stratified K-Fold Cross Validation

```
from sklearn.model_selection import StratifiedKFold
skf = StratifiedKFold(n_splits=5, shuffle=True, random_state=1)
f1_scores = []
#run for every split
for train_index, test_index in skf.split(normalised_train_df, y_balanced):
  x_train, x_test = np.array(normalised_train_df)[train_index],
                    np.array(normalised_train_df)[test_index]
  y_train, y_test  = y_balanced[train_index], y_balanced[test_index]
  model = LogisticRegression().fit(x_train, y_train)
  #save result to list
  f1_scores.append(f1_score(y_true=y_test, y_pred=model.predict(x_test), pos_label='2A'))
```

## Leave One Out Cross Validation (LOOCV)

```
from sklearn.model_selection import LeaveOneOut
loo = LeaveOneOut()
scores = cross_val_score(LogisticRegression(), normalised_train_df, y_balanced, cv=loo,
                         scoring='f1_macro')
average_score = scores.mean() * 100
```

## Confusion Matrix

```
from sklearn.metrics import recall_score, accuracy_score, precision_score, f1_score,
confusion_matrix
new_predictions = log_reg.predict(normalised_test_df)
cnf_mat = confusion_matrix(y_true=y_test, y_pred=new_predictions, labels=['2A', '3A'])
cnf_mat #prints       array([[  35, 34],
                             [  50, 58]])
```

## Accuracy

```
accuracy = accuracy_score(y_true=y_test, y_pred=new_predictions)
print('Accuracy: {}'.format(round(accuracy*100), 2)) #prints 53.0
```

## Precision

```
precision = precision_score(y_true=y_test, y_pred=new_predictions, pos_label='2A')
print('Precision: {}'.format(round(precision*100), 2)) #prints 41.0
```

## Recall

```
recall = recall_score(y_true=y_test, y_pred=new_predictions, pos_label='2A')
print('Recall: {}'.format(round(recall*100), 2)) #prints 51.0
```

## F1-Score

```
f1 = f1_score(y_true=y_test, y_pred=new_predictions, pos_label='2A')
print('F1: {}'.format(round(f1*100), 2)) #prints 45.0
```