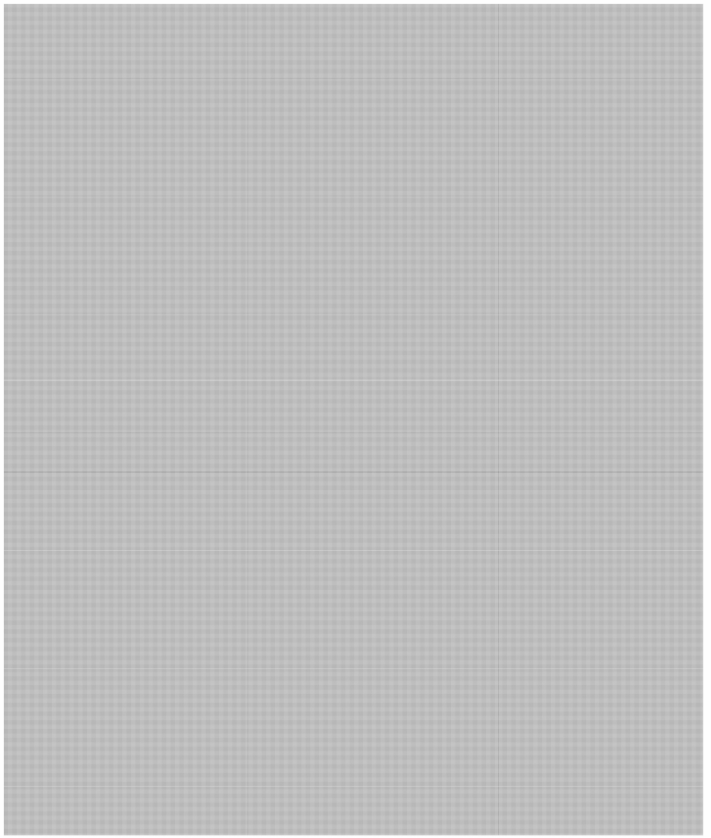
Bounding Boxes

s.19(1)

s.20(1)(b)

ProjectNumber	Lead	RecipientName	RecipientType	Indigenous	VulnerableGroups	Town_City_Community	Province_Territory	PostalCode	ContactName	ContactPhone	ContactEmail	Status
521-1	Breakfast Club of Canada	100 Mile Elementary	School/Educational Centre	No	Youth	100 Mile House	BC					
521-1	Breakfast Club of Canada	100 Mile Elementary	School/Educational Centre	No	Youth	100 Mile House	BC					
521-10	Breakfast Club of Canada	Angik School	School/Educational Centre	Inuit	Youth	Paulatuk	NT					
521-10	Breakfast Club of Canada	Angik School	School/Educational Centre	Inuit	Youth	Paulatuk	NT					
521-10	Breakfast Club of Canada	Angik School	School/Educational Centre	Inuit	Youth	Paulatuk	NT					
521-10	Breakfast Club of Canada	Angik School	School/Educational Centre	Inuit	Youth	Paulatuk	NT					
521-100	Breakfast Club of Canada	Lake Trail Middle School	School/Educational Centre	No	Youth	Courtenay	BC					
521-100	Breakfast Club of Canada	Lake Trail Middle School	School/Educational Centre	No	Youth	Courtenay	BC					
521-100	Breakfast Club of Canada	Lake Trail Middle School	School/Educational Centre	No	Youth	Courtenay	BC					
521-1000	Breakfast Club of Canada	St. Anthony Schooi	School/Educational Centre	No	Youth	Calgary	AB					
521-1000	Breakfast Club of Canada	St. Anthony School	School/Educational Centre	No	Youth	Calgary	AB					



000057

Funding Status Classification using Machine Learning

Data Preprocessing A

```
#feature-transformation

# Define the transformation function
def transform_funding_status(x):
    if x in ['Approved', 'Approved-Other']:
        return 'Approved'
    else:
        return 'NotApproved'

# Apply the transformation to the 'Funding_Status' column
data['Funding_Status'] = data['Funding_Status'].apply(lambda x: transform_funding_status(x))

# Display the updated DataFrame values for Funding_Status
data.Funding_Status.value_counts()
```

```
Approved       3255
NotApproved     258
Name: Funding_Status, dtype: int64
```

```
#downsampling


# Count the number of 'Approved' and 'NotApproved' rows in the 'Funding_Status' column
counts = data['Funding_Status'].value_counts()

# Determine the size of the downsampled 'Approved' subset
n_approved = counts['Approved']
n_not_approved = counts['NotApproved']
downsampled_size = n_not_approved

# Randomly select a subset of the 'Approved' rows to keep
approved_subset = data[data['Funding_Status'] == 'Approved'].sample(n=downsampled_size)

# Select all the 'Not Approved' rows
not_approved_subset = data[data['Funding_Status'] == 'NotApproved']

# Concatenate the 'Approved' and 'Not Approved' subsets into a single DataFrame
downsampled_data = pd.concat([approved_subset, not_approved_subset])

# Shuffle the rows of the downsampled DataFrame
downsampled_data = downsampled_data.sample(frac=1, random_state=42)
downsampled_data.shape
```

```
(516, 5)
```

```
downsampled_data.Funding_Status.value_counts()
```

```
NotApproved    258
Approved       258
Name: Funding_Status, dtype: int64
```

Data Preprocessing B

```
X = downsampled_data[['Lead', 'Town_City_Community', 'Approved_Funding', 'Investment_Type']]
y = downsampled_data['Funding_Status']
```

```
#preprocessing
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.impute import SimpleImputer

# Encode categorical variables
le = LabelEncoder()
X['Lead'] = le.fit_transform(X['Lead'])
X['Town_City_Community'] = le.fit_transform(X['Town_City_Community'])
X['Investment_Type'] = le.fit_transform(X['Investment_Type'])

# Handle missing values
imputer = SimpleImputer()
X = imputer.fit_transform(X)

# Scale the features
scaler = StandardScaler()
X = scaler.fit_transform(X)
```

Model Training

```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25, random_state=42)
```

```
from sklearn.linear_model import LogisticRegression

# Train the model
clf = LogisticRegression()
clf.fit(X_train, y_train)

# Make predictions
y_pred = clf.predict(X_test)
```

Model Evaluation

```
# Evaluate the model
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score,confusion_matrix,classification_report

print('Accuracy:', accuracy_score(y_test, y_pred))
print('F1 Score:', f1_score(y_test, y_pred,pos_label='Approved',average='micro'))
```

```
Accuracy: 0.8372093023255814
F1 Score: 0.8372093023255814
/usr/local/lib/python3.9/dist-packages/sklearn/metrics/_classification.py:1396: UserWarning: Note that pos_label (set to 'Approved')
  warnings.warn(
```

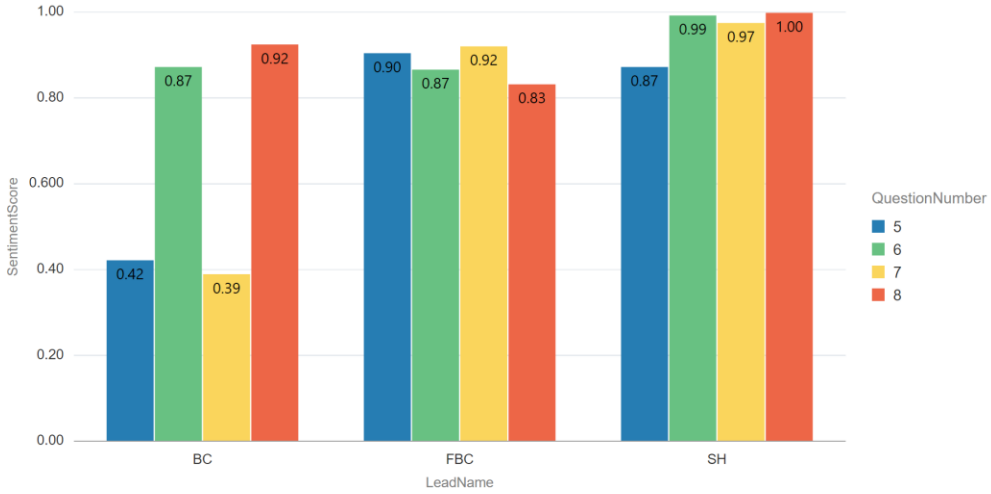
```
print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

    Approved       0.98      0.72      0.83        71
  NotApproved       0.74      0.98      0.84        58

    accuracy                           0.84       129
   macro avg       0.86      0.85      0.84       129
weighted avg       0.87      0.84      0.84       129
```

Sentiment Analysis

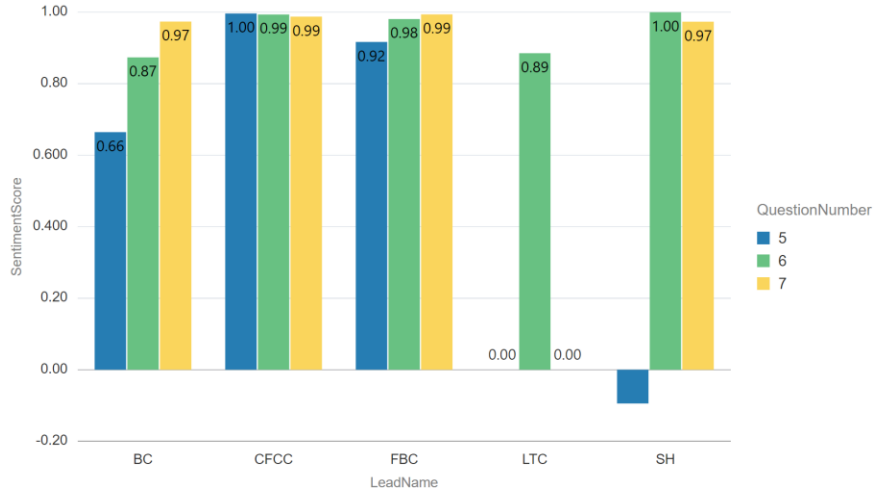
SentimentScore by LeadName & FundType of Feedback Form

FundType: EF



SentimentScore by LeadName & FundType of Feedback Form

QuestionNumber: 5, 6, 7 FundType: LIF



Sentiment Score of Second Harvest LIF FundType for Q5

QuestionNumber: 5 LeadName: SH FundType: LIF

FeedbackText	SentimentScore
Despite our best efforts to avoid duplication of funding, this was the biggest challenge during the granting period. Second Harvest developed a National Directory that was posted to all national delivery partners online (Food Banks Canada, Community Food Centres Canada, Salvation Army, Breakfast Clubs of Canada), which was to be updated on a regular basis by all organizations to identify areas where duplication of funding may occur. Our funding application also included a series of eligibility questions for the applicant organizations, which included asking the organizations to self-identify whether they were a member of, or applied for funding from, any of the other national delivery partners. These tactics worked to an extent, however, manually updating this information was cumbersome for the national delivery partners and without timely updates from all participants, we were unable to ensure 100% of the time that all duplicate funding requests were identified in advance of funds being released to the applicants.	-0.09

Sentiment Scoring Sample Text