Predicting Type 2 Diabetes Using Machine Learning

Project Members: Adam Dunn, Kelvin Idwe, Jeremy Pimentel, Mohamed Yaakoub

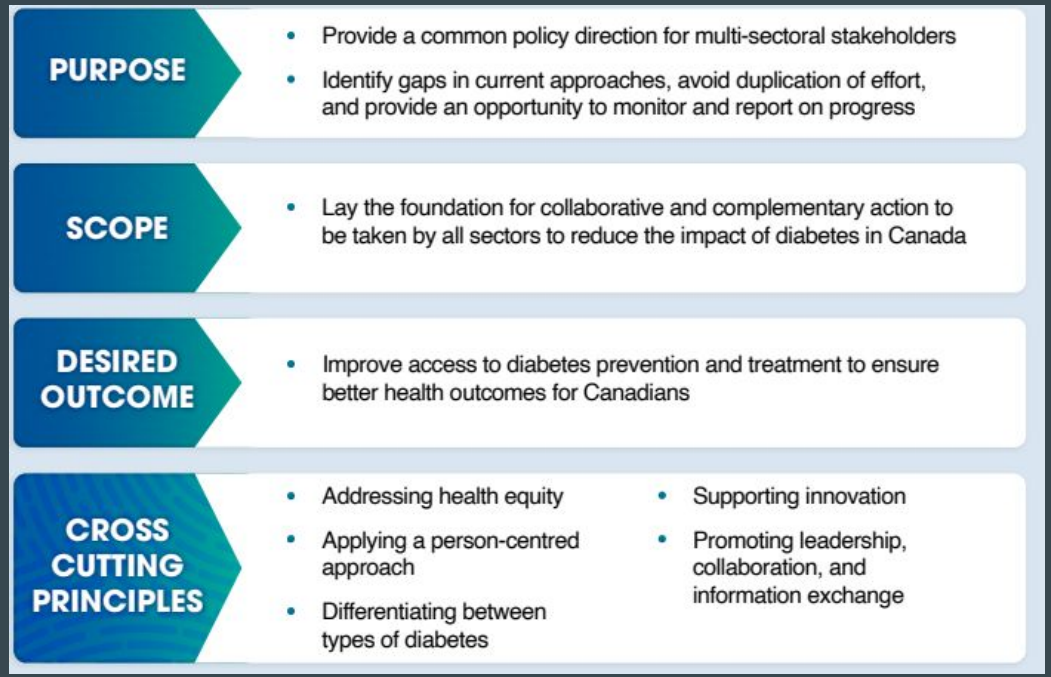
Why Diabetes?

https://www.diabetes.ca/DiabetesCanadaWebsite/media/Campaigns/NDAM/Type-2-diabetes-risk-factors-infographic_FNL.pdf

Developing Tools to Identify Type - 2 Diabetes Risk Important

From Canada's Framework on Handling Diabetes

(https://www.canada.ca/content/dam/phac-aspc/documents/services/publications/diseases-conditions/framework-diabetes-canada-infographic/framework-diabetes-canada-infographic-en.pdf)



Questions We Hope to Answer

What variables are key identifiers in determining type 2 diabetes?

Can a machine learning model accurately identify type 2 diabetes?

Can this tool be used in Canada’s plan to tackle diabetes?

Source of Data

CDC - https://www.cdc.gov/Nchs/Nhanes/about_nhanes.htm

NHANES Data Release and Access Policy - Public Release on a bi-anual basis.

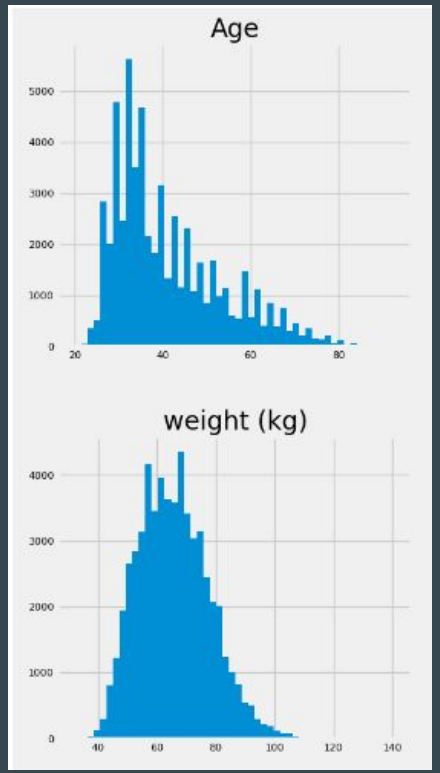
Data Exploration

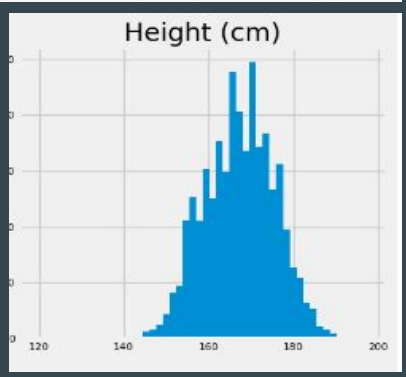
Data Exploration

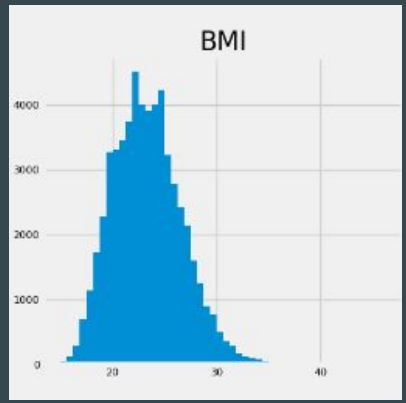
```
## EDA
df.info()
```

```
Int64Index: 57595 entries, 0 to 202392
Data columns (total 11 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   Age                       57595 non-null  int64
 1   Gender                    57595 non-null  int64
 2   Height (cm)               57595 non-null  float64
 3   weight (kg)               57595 non-null  float64
 4   Outcome (Type 2 Diabetes) 57595 non-null  int64
 5   Smokes                    57595 non-null  float64
 6   Drinks Alcohol            57595 non-null  float64
 7   Family History of Diabetes 57595 non-null  int64
 8   BMI                       57595 non-null  float64
 9   Cholesterol               57595 non-null  float64
 10  Fasting Plasma Glucose    57595 non-null  float64
dtypes: float64(7), int64(4)
memory usage: 5.3 MB
```

```
df.hist(figsize=(16, 20), bins=50, xlabelsize=8, ylabelsize=8);
```







Data Analysis Phase

Data Analysis

Data Preprocessing

diabetes	smokerid	drinkerid	f
no	never smoker	never drinker	
no	current smoker	former drinker	
no	never smoker	never drinker	
no	never smoker	never drinker	
no	never smoker	never drinker	
no	never smoker	former drinker	

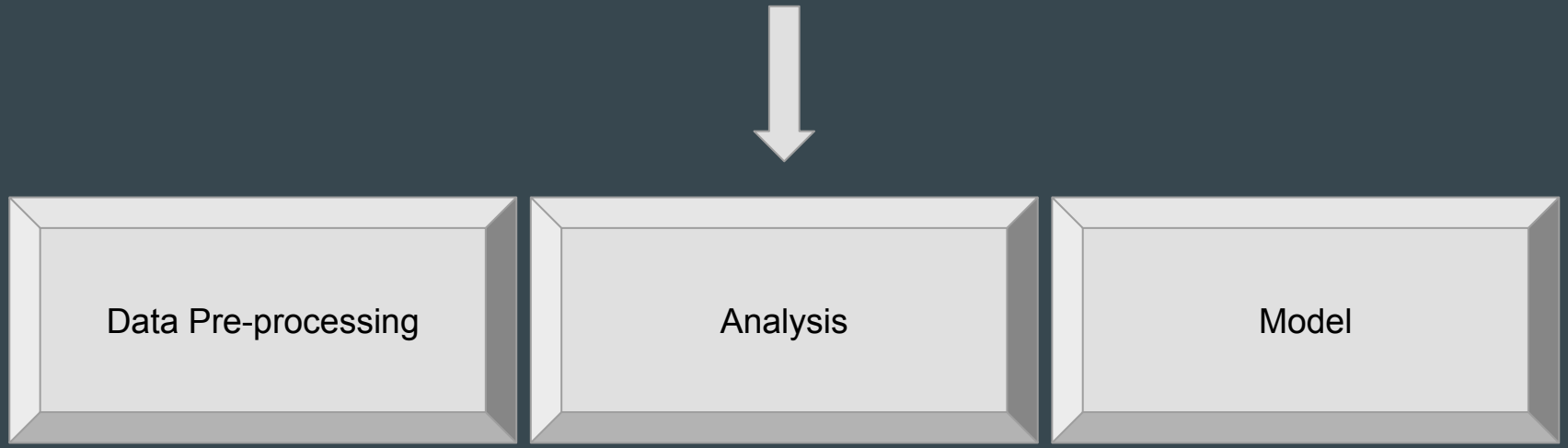
diabetes	smokerid	drinkerid
0	0	0
0	1	2
0	0	0
0	0	0
0	0	0
0	0	2
0	1	0
0	1	1

Converting string values to numeric values

Dropping rows with ‘no info’ and unnecessary columns

```
final_results_df2 = final_results_df[final_results_df.smokerid != 'no info']
final_results_df3 = final_results_df2[final_results_df2.drinkerid != 'no info']
final_results_df = final_results_df3.drop(['personid'], axis=1)
final_results_df.head(20)
```

Data Analysis



Data Analysis

- Descriptive statistics of our dataset.

- Helps us to identify basic features of the data.

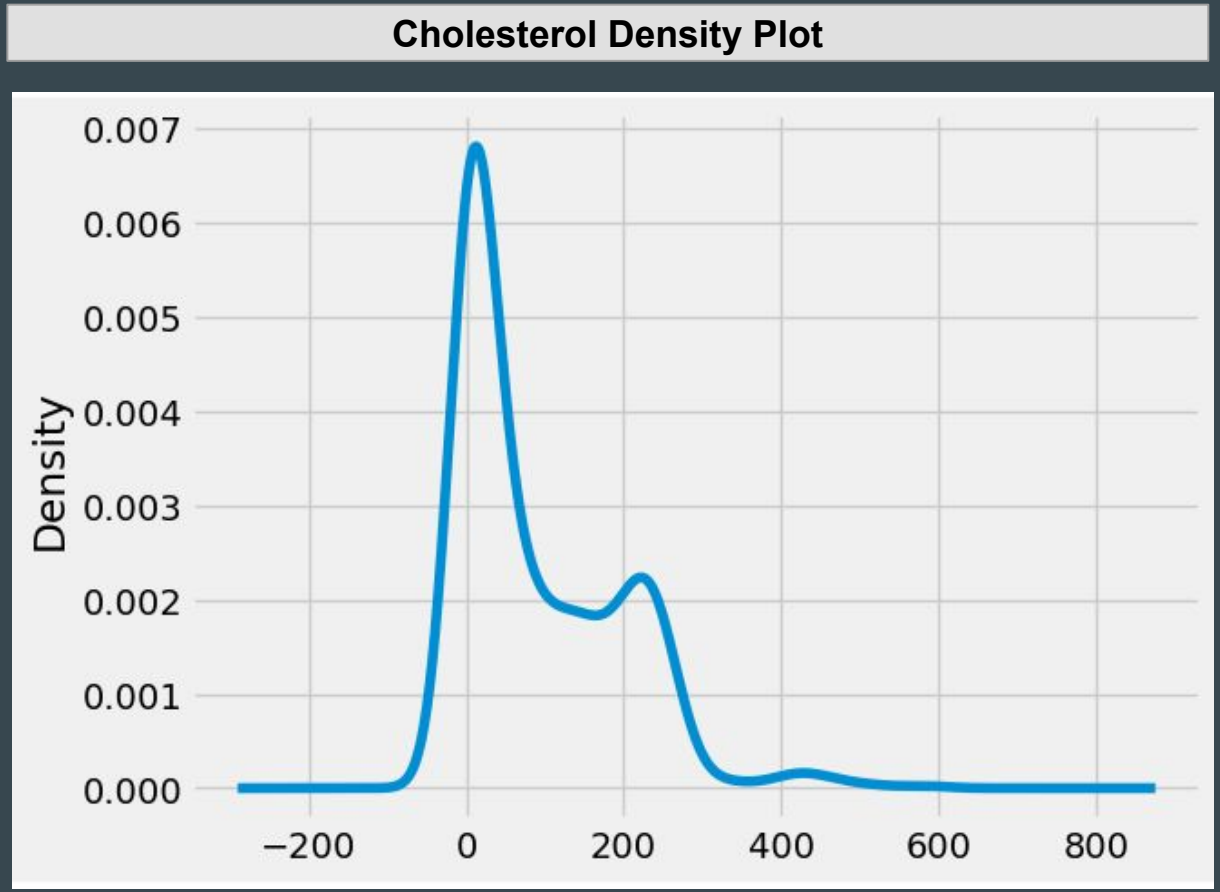
```
X = final_results_df.drop("diabetes",axis = 1)
Y = final_results_df['diabetes']
X.describe().T
```

	count	mean	std	min	25%	50%	75%	max
age	57595.0	41.306138	12.444804	20.00	32.000000	37.000000	48.000000	93.00
sexid	57595.0	0.651723	0.476428	0.00	0.000000	1.000000	1.000000	1.00
height	57595.0	167.314262	8.235737	119.30	161.000000	167.900000	173.000000	198.00
weight	57595.0	65.722785	12.159178	32.00	56.500000	65.000000	73.700000	141.00
bmi	57595.0	23.369591	3.321500	15.00	20.900000	23.200000	25.500000	46.30
sbp	57595.0	118.893411	14.817999	72.00	108.000000	118.000000	129.000000	168.00
dbp	57595.0	74.368730	9.887784	43.00	67.000000	74.000000	81.000000	105.00
fpg	57595.0	4.963058	0.618607	1.99	4.570000	4.970000	5.350000	6.99
cholesterol	57595.0	4.695943	0.871020	0.04	4.080000	4.620000	5.230000	11.65
triglyceride	57595.0	1.402390	1.046501	0.00	0.780000	1.130000	1.690000	32.64
hdl	57595.0	1.350319	0.215957	0.48	1.280000	1.372496	1.372496	2.29
ldl	57595.0	2.740484	0.491419	0.79	2.570000	2.763127	2.770000	4.79
alt	57595.0	25.099754	21.887497	1.00	13.400000	19.100000	28.900000	722.30
ast	57595.0	24.410476	6.546769	0.00	23.992542	23.992542	23.992542	260.90
bun	57595.0	4.649935	1.115462	0.90	3.900000	4.652664	5.260000	15.42
ccr	57595.0	71.751725	14.829545	27.70	60.500000	71.600000	82.000000	274.20
fpg_finalvisit	57595.0	5.167683	0.648026	3.25	4.800000	5.100000	5.410000	20.60
smokerid	57595.0	0.283861	0.537450	0.00	0.000000	0.000000	0.000000	2.00
drinkerid	57595.0	0.323830	0.721404	0.00	0.000000	0.000000	0.000000	2.00
famhistid	57595.0	0.050560	0.219099	0.00	0.000000	0.000000	0.000000	1.00

Data Analysis

- Created density plots for features.

- Showcases distribution of values for our model, similar to the histograms.



Data Analysis

- Showed us a lot more people in this dataset don't have type-2 diabetes.

- Great for them, hurt our original model choice.

Value Count of Diabetes by Gender

```
Gender  Outcome (Type 2 Diabetes)
0       0                         36681
        1                           855
1       0                         19837
        1                           222
Name: Outcome (Type 2 Diabetes), dtype: int64
```

Gender 0 = Male
Gender 1 = Female

Outcome 0 = No Diabetes
Outcome 1 = Diabetes

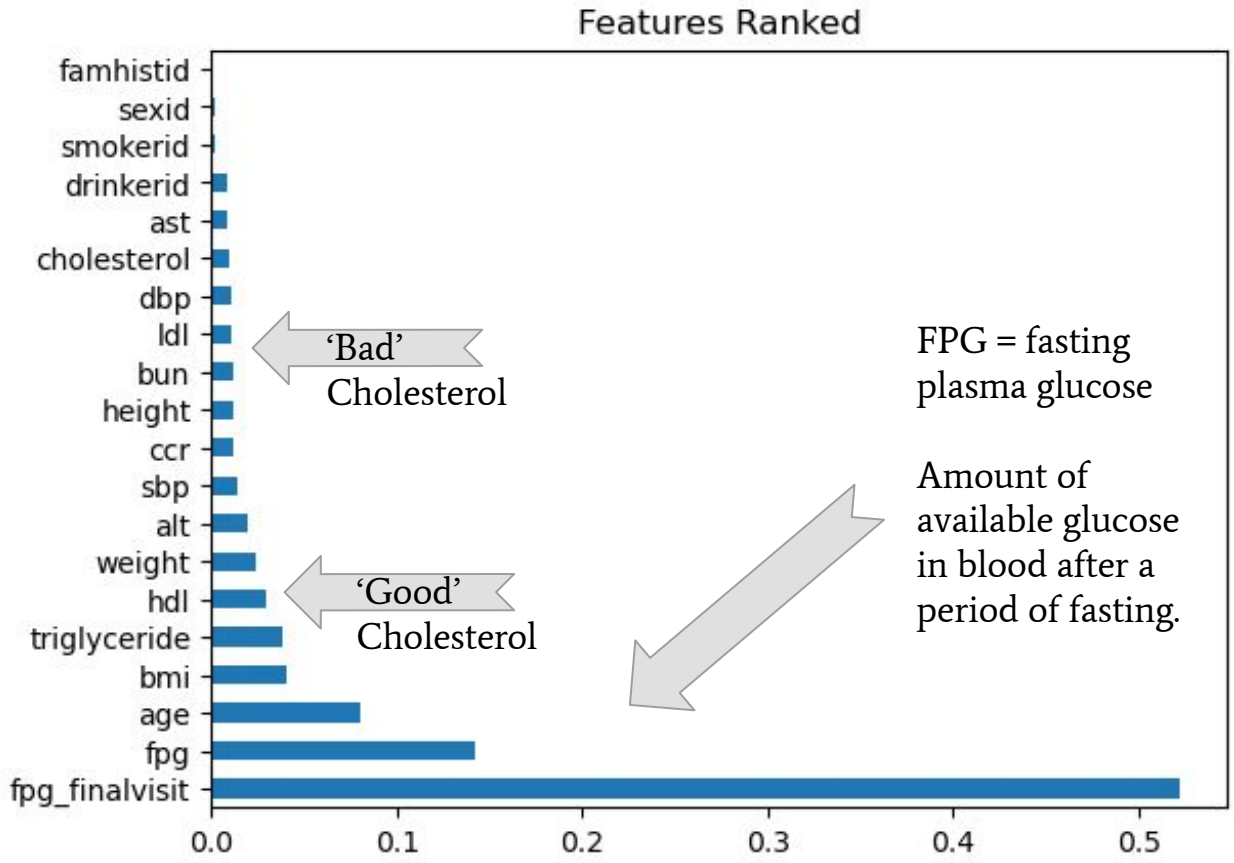
Data Analysis

Model

- We started with a KNN model (K- Nearest Neighbors Algorithm).
    - Simple to implement, but has issues with unbalanced data.
- Had low accuracy score for our purpose (classifying diabetes).

- Used **Balanced Random Forest Classifier** instead with train_test_split() to train the model.
- Also used Undersampling Cluster Centroids since our data was heavily skewed towards people without diabetes.

Feature Selection / Engineering


Features Ranked
famhistid
sexid
smokerid
drinkerid
ast
cholesterol
dbp
ldl
bun
height
ccr
sbp
alt
weight
hdl
triglyceride
bmi
age
fpg
fpg_finalvisit
0.0
0.1
0.2
0.3
0.4
0.5
'Bad' Cholesterol
'Good' Cholesterol
FPG = fasting plasma glucose
Amount of available glucose in blood after a period of fasting.

Results of Model

Results of Model - KNN (Original Model)

Accuracy Score: 76.8%

```
Confusion matrix:
 [[2493  727]
 [  33   28]]
accuracy_score:
 0.7683633038707711
classification_report:
              precision    recall  f1-score   support

           0       0.99      0.77      0.87      3220
           1       0.04      0.46      0.07        61

    accuracy                           0.77      3281
   macro avg       0.51      0.62      0.47      3281
weighted avg       0.97      0.77      0.85      3281
```

Results of Model - Random Forest Classifier

Accuracy Score: 98.6%

```
Confusion matrix:
 [[11187   110]
 [   48   174]]
accuracy_score:
 0.9862835315565587
```

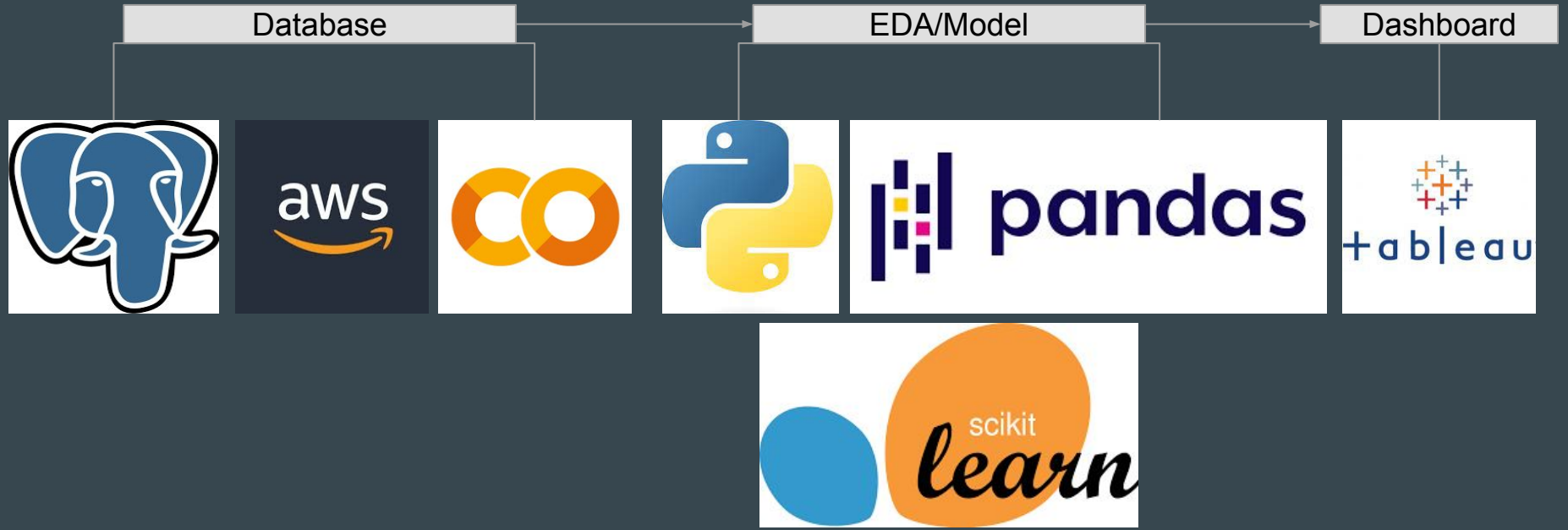
With SMOTE

```
# Print the imbalanced classification report
from imblearn.metrics import classification_report_imbalanced
print(classification_report_imbalanced(y_test, y_pred))
```

✓ 0.3s

	pre	rec	spe	f1	geo	iba	sup
0	1.00	0.99	0.75	0.99	0.87	0.77	3218
1	0.72	0.75	0.99	0.74	0.87	0.73	65
avg / total	0.99	0.99	0.76	0.99	0.87	0.77	3283

Technologies Used



Further Recommendations/Things We Would Change

- We would recommend developing a model from the ground up that can continuously be trained with new data.

- We would continue to experiment with feature selection to determine the most import features for the model.

- If we started again, we would try and gather more data earlier on and try to code a dashboard with JS/HTML for the experience.