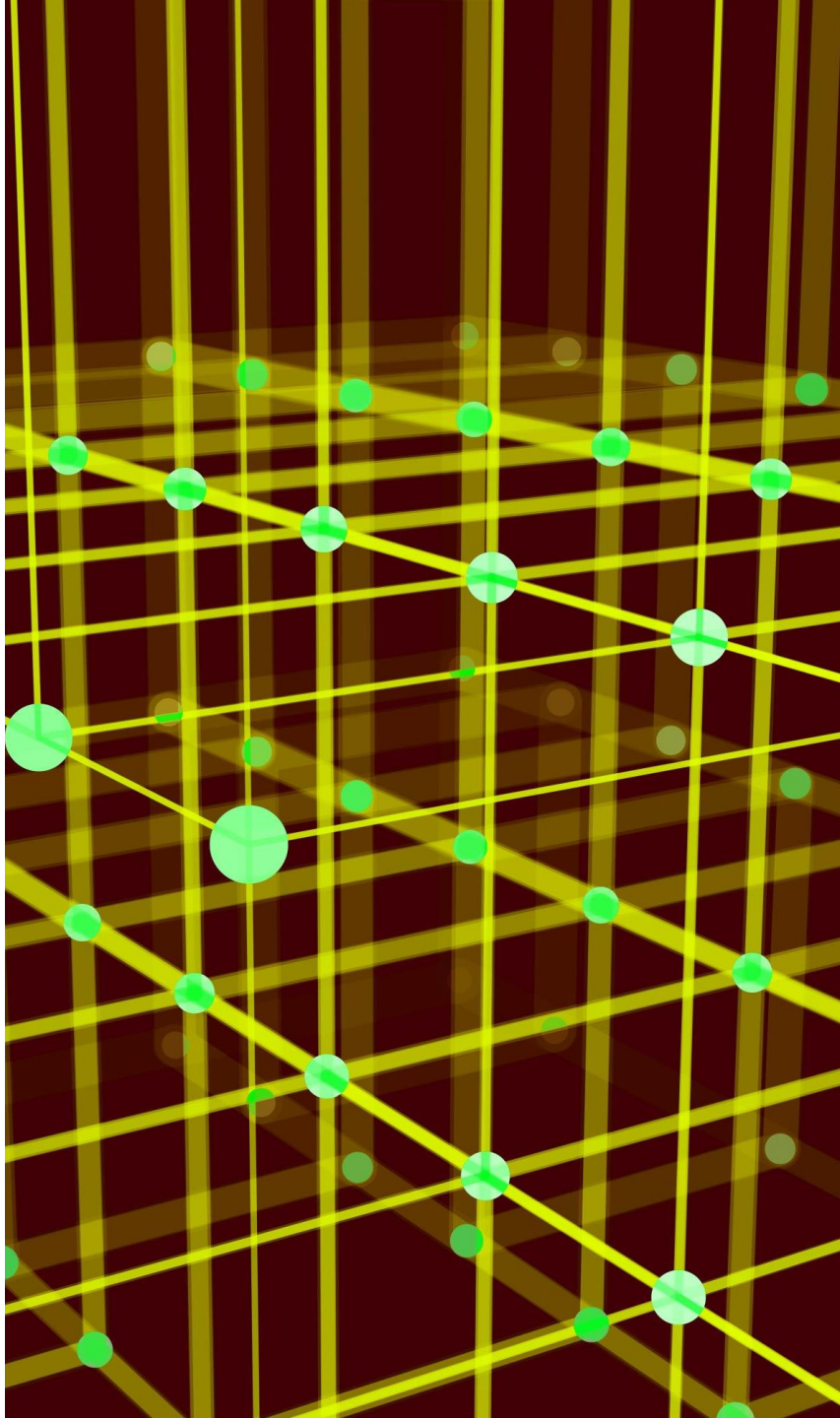
NAÏVE BAYES

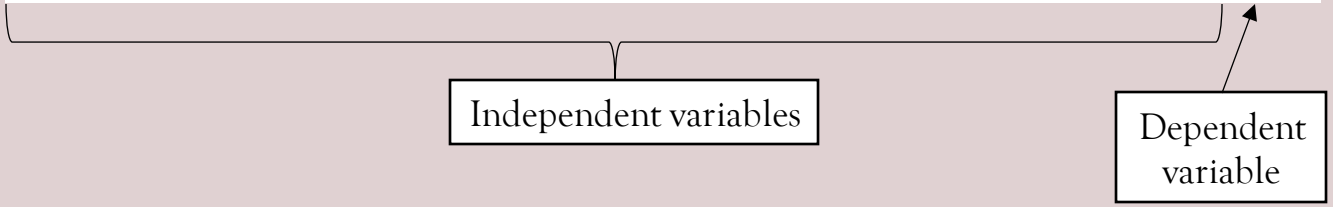
Azure Eller, Hayden Muscha,

Nathan Van Schyndel, Victor Pham



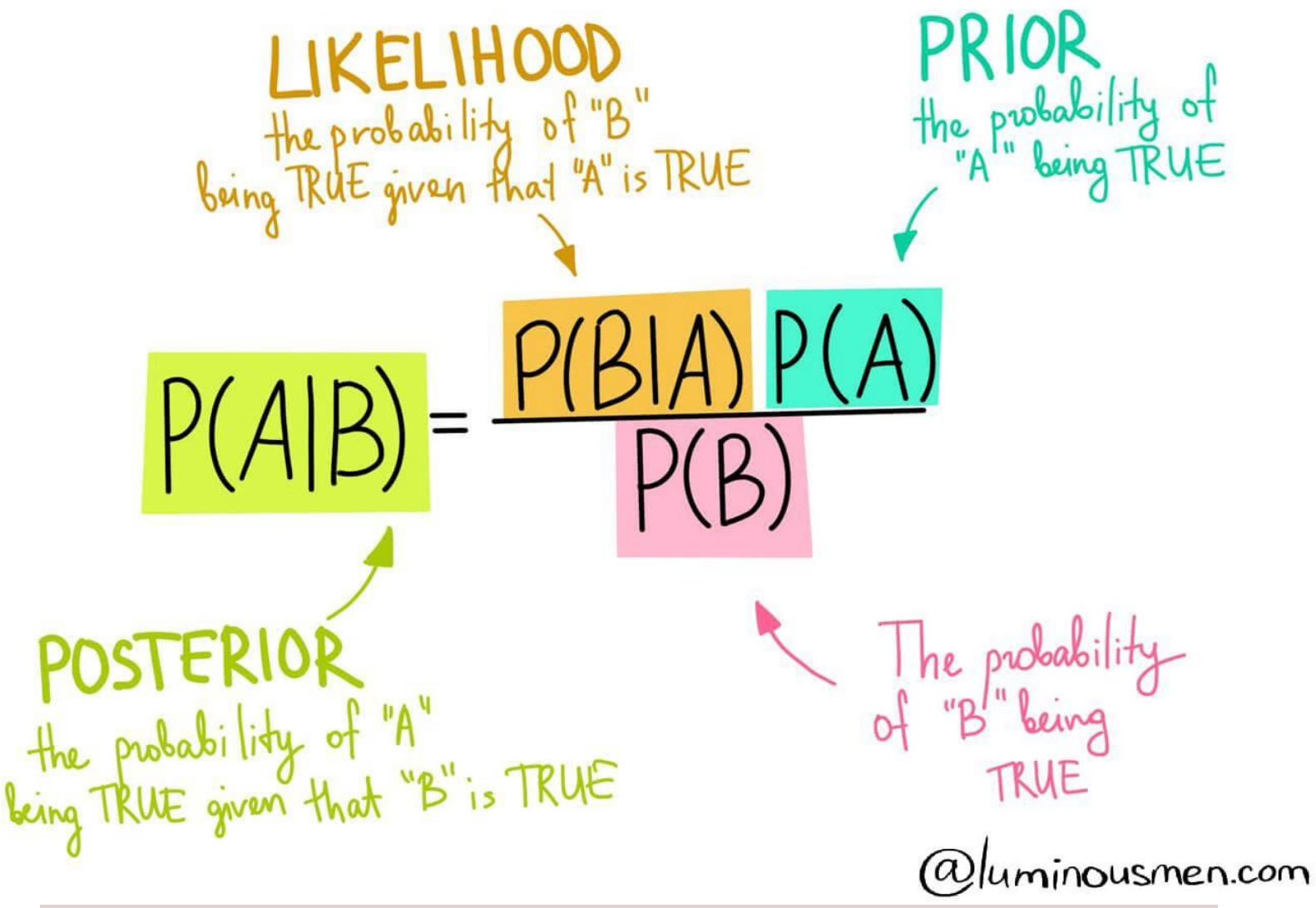
DATA SET

Ethnicity	YearsEmployed	PriorDefault	Employed	CreditScore	DriversLicense	Citizen	ZipCode	Income	Approved
White	1.25	1	1	1	0	ByBirth	202	0	1
Black	3.04	1	1	6	0	ByBirth	43	560	1
Black	1.50	1	0	0	0	ByBirth	280	824	1
White	3.75	1	1	5	1	ByBirth	100	3	1
White	1.71	1	0	0	0	ByOtherMeans	120	0	1



- Credit Card Approvals
- Small Data Set ~ 600 entries
- Mix of categorical (Boolean) and continuous fields

NAÏVE BAYES


LIKELIHOOD
the probability of "B" being TRUE given that "A" is TRUE
PRIOR
the probability of "A" being TRUE
P(A|B) = P(B|A) P(A) / P(B)
POSTERIOR
the probability of "A" being TRUE given that "B" is TRUE
The probability of "B" being TRUE
@luminousmen.com

GAUSSIAN NB

```
X = credit.copy().drop(columns=['Gender','Approved','Married','BankCustomer',
    'Industry','Ethnicity','PriorDefault','Employed','DriversLicense','Citizen','ZipCode'])
y = credit['Approved'].copy()
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25, random_state=0)
```

```
gnb = GaussianNB()
#priors = (.3,.7)
y_pred = gnb.fit(X_train, y_train).predict(X_test)
print("Number of mislabeled points out of a total %d points : %d"
% (X_test.shape[0], (y_test != y_pred).sum()))
gnb.fit(X_train, y_train)
print(gnb)
# make predictions
expected = y_test
predicted = gnb.predict(X_test)
# summarize the fit of the model
print(metrics.classification_report(expected, predicted))
print(metrics.confusion_matrix(expected, predicted))
print(gnb.score(X_test, y_test))
noyes = list(gnb.class_prior_)
```

```
Number of mislabeled points out of a total 173 points : 52
GaussianNB()
              precision    recall  f1-score   support

           0       0.66      0.94      0.77        94
           1       0.85      0.42      0.56        79

    accuracy                           0.70       173
   macro avg       0.75      0.68      0.67       173
weighted avg       0.74      0.70      0.67       173

[[88  6]
 [46 33]]
0.6994219653179191
```

```
Number of mislabeled points out of a total 173 points : 50
GaussianNB(priors=(0.3, 0.7))
              precision    recall  f1-score   support

           0       0.67      0.91      0.77        94
           1       0.82      0.47      0.60        79

    accuracy                           0.71       173
   macro avg       0.75      0.69      0.69       173
weighted avg       0.74      0.71      0.69       173

[[86  8]
 [42 37]]
0.7109826589595376
```

INCREASING MODEL PERFORMANCE GAUSSIAN

CATEGORICAL NB

```
credit = credit.drop(columns = ['Age','Debt','YearsEmployed','CreditScore','ZipCode','Income'])
credit = pd.get_dummies(credit,drop_first=True)
X = credit.copy().drop(columns=['Approved'])
y = credit['Approved'].copy()
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25, random_state=0)
```

```
cat = CategoricalNB()
y_pred = cat.fit(X_train, y_train).predict(X_test)
print("Number of mislabeled points out of a total %d points : %d"
% (X_test.shape[0], (y_test != y_pred).sum()))
cat.fit(X_train, y_train)
print(cat)
# make predictions
expected = y_test
predicted = cat.predict(X_test)
# summarize the fit of the model
print(metrics.classification_report(expected, predicted))
print(metrics.confusion_matrix(expected, predicted))
print(cat.score(X_test, y_test))
```

```
Number of mislabeled points out of a total 173 points : 34
CategoricalNB()
              precision    recall  f1-score   support

           0       0.80      0.85      0.82        94
           1       0.81      0.75      0.78        79

    accuracy                           0.80       173
   macro avg       0.80      0.80      0.80       173
weighted avg       0.80      0.80      0.80       173

[[80 14]
 [20 59]]
0.8034682080924855
```

```
Number of mislabeled points out of a total 173 points : 30
CategoricalNB(class_prior=(0.4, 0.6))
              precision    recall  f1-score   support

           0       0.86      0.82      0.84        94
           1       0.80      0.84      0.81        79

    accuracy                           0.83       173
   macro avg       0.83      0.83      0.83       173
weighted avg       0.83      0.83      0.83       173

[[77 17]
 [13 66]]
0.8265895953757225
```

INCREASING MODEL PERFORMANCE CATEGORICAL

Number of mislabeled points out of a total 173 points : 43

	precision	recall	f1-score	support
0	0.70	0.94	0.80	94
1	0.88	0.53	0.66	79
accuracy			0.75	173
macro avg	0.79	0.73	0.73	173
weighted avg	0.78	0.75	0.74	173

```
[[88  6]
 [37 42]]
0.7514450867052023
```

LOGISTIC REGRESSION

Number of mislabeled points out of a total 173 points : 27

	precision	recall	f1-score	support
0	0.85	0.86	0.86	94
1	0.83	0.82	0.83	79
accuracy			0.84	173
macro avg	0.84	0.84	0.84	173
weighted avg	0.84	0.84	0.84	173

```
[[81 13]
 [14 65]]
0.8439306358381503
```

Comparative Models:

Logistic Regression

&

Random Forest Classifier

RandomForestClassifier()

Number of mislabeled points out of a total 173 points : 45

	precision	recall	f1-score	support
0	0.74	0.81	0.77	94
1	0.74	0.66	0.70	79
accuracy			0.74	173
macro avg	0.74	0.73	0.73	173
weighted avg	0.74	0.74	0.74	173

```
[[76 18]
 [27 52]]
0.7398843930635838
```

RANDOM FOREST

RandomForestClassifier()

Number of mislabeled points out of a total 173 points : 26

	precision	recall	f1-score	support
0	0.83	0.91	0.87	94
1	0.88	0.77	0.82	79
accuracy			0.85	173
macro avg	0.86	0.84	0.85	173
weighted avg	0.85	0.85	0.85	173

```
[[86  8]
 [18 61]]
0.8497109826589595
```

CONCLUSIONS

- Adjusting Priors can increase performance of the model
- Best to run GaussianNB on continuous data, and CategoricalNB on categorical data
- More data would massively improve the performance of the model