# Naïve Bayes

Naïve Bayes Classifier

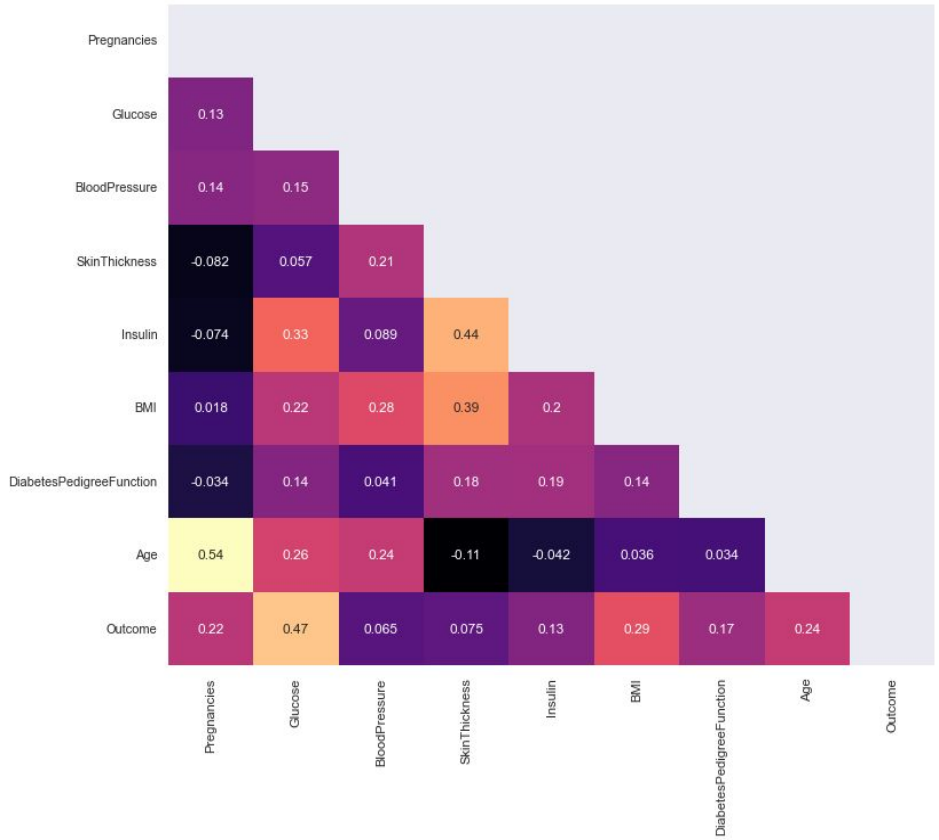
$$P(A|B) = \frac{P(B|A)\ P(A)}{P(B)}$$

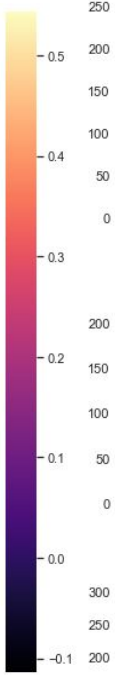
Thomas Bayes
1702 - 1761

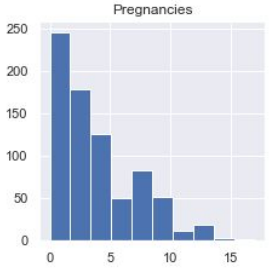
By: Kylee LaPierre, Beth Vander Hoek, Stephanie Leiva, & Dennis Kelly

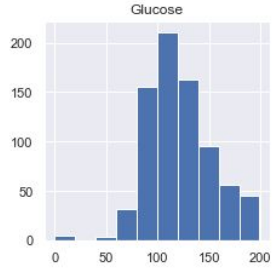
# Implementing Naïve Bayes

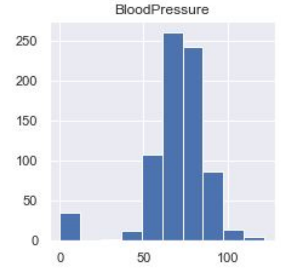
- **Pick a dataset requirements:**
  - Familiar/something we've used before
  - Categorical
  - TP, FP, TN, FN
  - **We chose the diabetes dataset**
- **Data processing steps:**
  - Pop out the dependent/outcome variable
  - Normal cleaning steps: nulls, outliers, type, zeros, correlation
  - Imputed 0's with mean vs. deleting
- **Train test split:** 25% testing
- **Other steps to implement model:**
  - Naive Bayes vs Logistic Regression Model Comparison
  - Confusion Matrix
  - ROC AUC

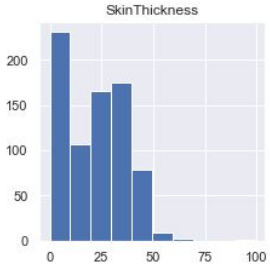

Pregnancies
Glucose
BloodPressure
SkinThickness
Insulin
BMI
DiabetesPedigreeFunction
Age
Outcome
0.13
0.14 0.15
-0.082 0.057 0.21
-0.074 0.33 0.089 0.44
0.018 0.22 0.28 0.39 0.2
-0.034 0.14 0.041 0.18 0.19 0.14
0.54 0.26 0.24 -0.11 -0.042 0.036 0.034
0.22 0.47 0.065 0.075 0.13 0.29 0.17 0.24

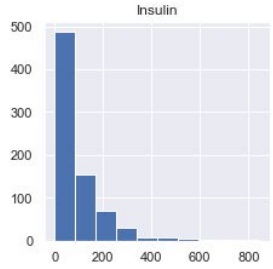

0.5
0.4
0.3
0.2
0.1
0.0
−0.1

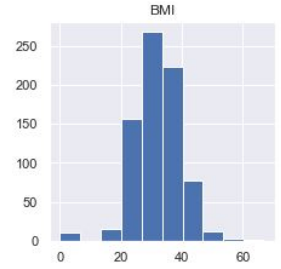

Pregnancies

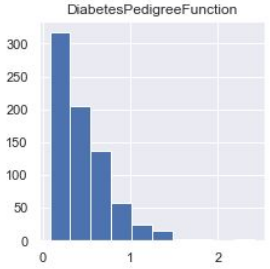

Glucose

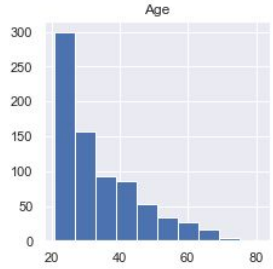

BloodPressure

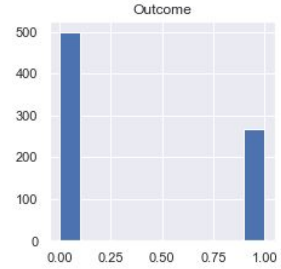

SkinThickness


Insulin


BMI

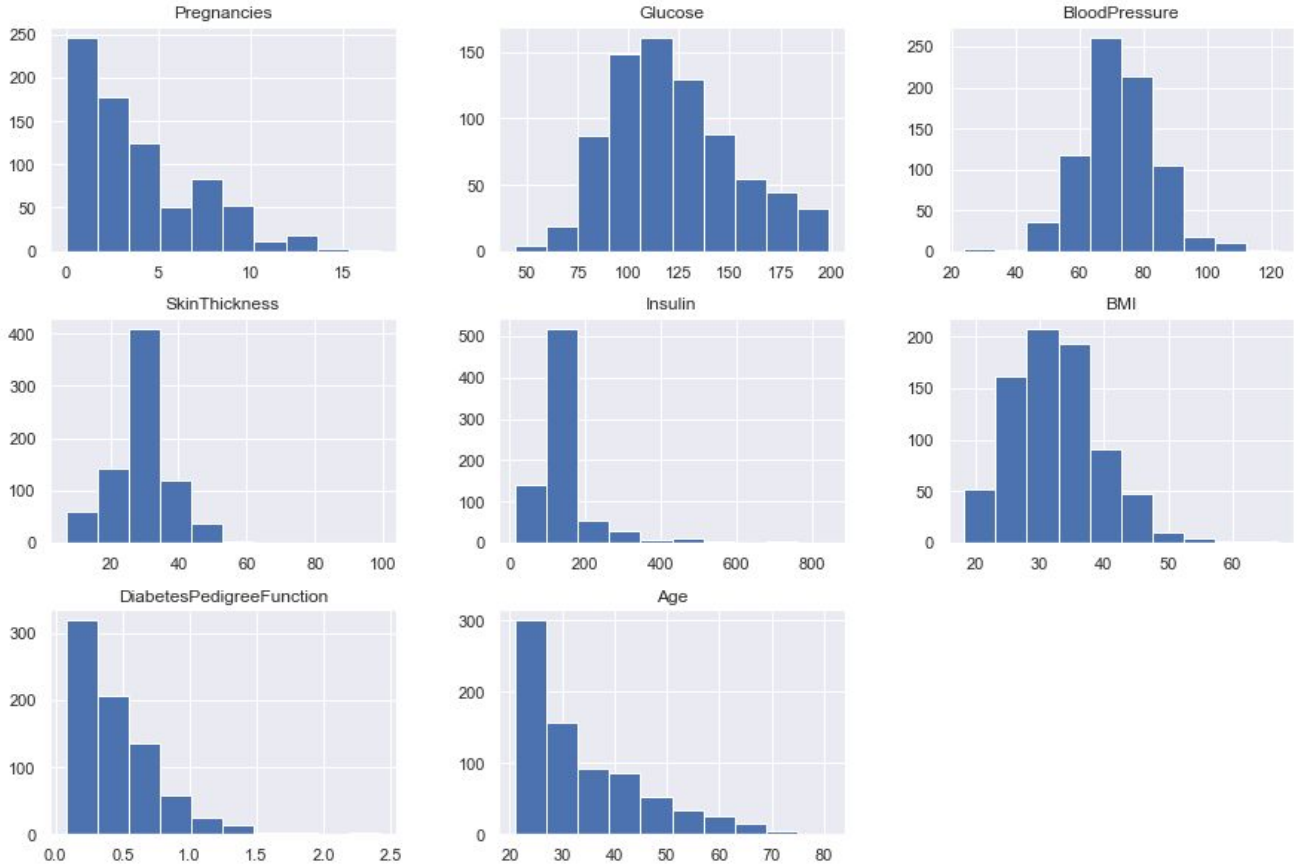

DiabetesPedigreeFunction


Age


Outcome

# Naïve Bayes Data Processing Code

```
diabetes = pd.read_csv("diabetes.csv")
diabetes.head(10)
diabetes.info()
diabetes.hist(figsize=(15, 10))
diabetes.describe()

outcome = diabetes.pop("Outcome")
imp = SimpleImputer(missing_values=0)

imputed_diabetes = diabetes.copy()
imputed_diabetes[['Glucose', 'BloodPressure', 'SkinThickness', 'Insulin', 'BMI']] = imp.fit_transform(diabetes[['Glucose', 'BloodPressure',
'SkinThickness', 'Insulin', 'BMI']])

imputed_diabetes.hist(figsize=(15, 10))
```


Pregnancies
Glucose
BloodPressure
SkinThickness
Insulin
BMI
DiabetesPedigreeFunction
Age

# Naïve Bayes Implementation Code

```
1  X_train, X_test, y_train, y_test = train_test_split(imputed_diabetes, outcome, test_size=0.25)
2  gnb = GaussianNB()
3
4  gnb.fit(X_train, y_train)
5  print("Naive Bayes Training Summary")
6  print("============================")
7  print(f"Means: {gnb.theta_}")
8  print(f"Variances: {gnb.var_}")
9  print(f"In-sample accuracy: {gnb.score(X_train, y_train):.3f}")
```

✓ 0.4s

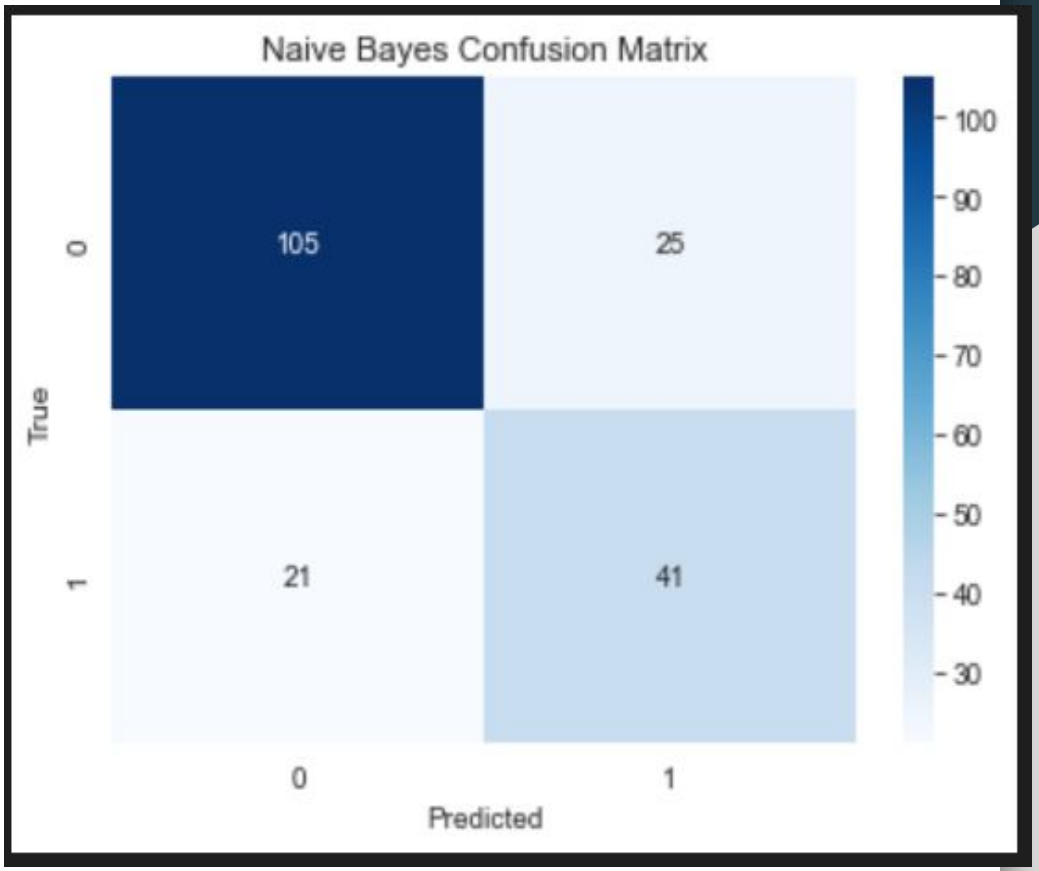
# Naïve Bayes Hyperparameters Tuning Code

- For our dataset + GaussianNB(), hyperparameter tuning did not improve accuracy
- Var_smoothing unneeded since we did not have underrepresented outcomes
- Priors tuning did not help model accuracy
    - Either can increase number of true positives (& miss negatives) or increase number of true negatives (& miss positives)

# Model Evaluation

```
Naive Bayes Training Summary
============================
Means: [[  3.28947368 110.36414086  70.91649314  27.99279599 139.75917713
   30.87049294   0.43781316  31.36052632]
 [  4.80102041 141.42697328  74.59113372  31.80519823 179.86315135
   35.052334     0.54401531  36.41326531]]
Variances: [[9.42147542e+00 5.95267832e+02 1.45234268e+02 7.58526394e+01
  5.31690916e+03 4.34272115e+01 9.53376903e-02 1.39641081e+02]
 [1.28226593e+01 8.53980972e+02 1.40699146e+02 8.25059028e+01
  1.00110031e+04 4.42340260e+01 1.51685423e-01 1.07875137e+02]]
In-sample accuracy: 0.760
Out of sample, test accuracy: 0.7604166666666666
```

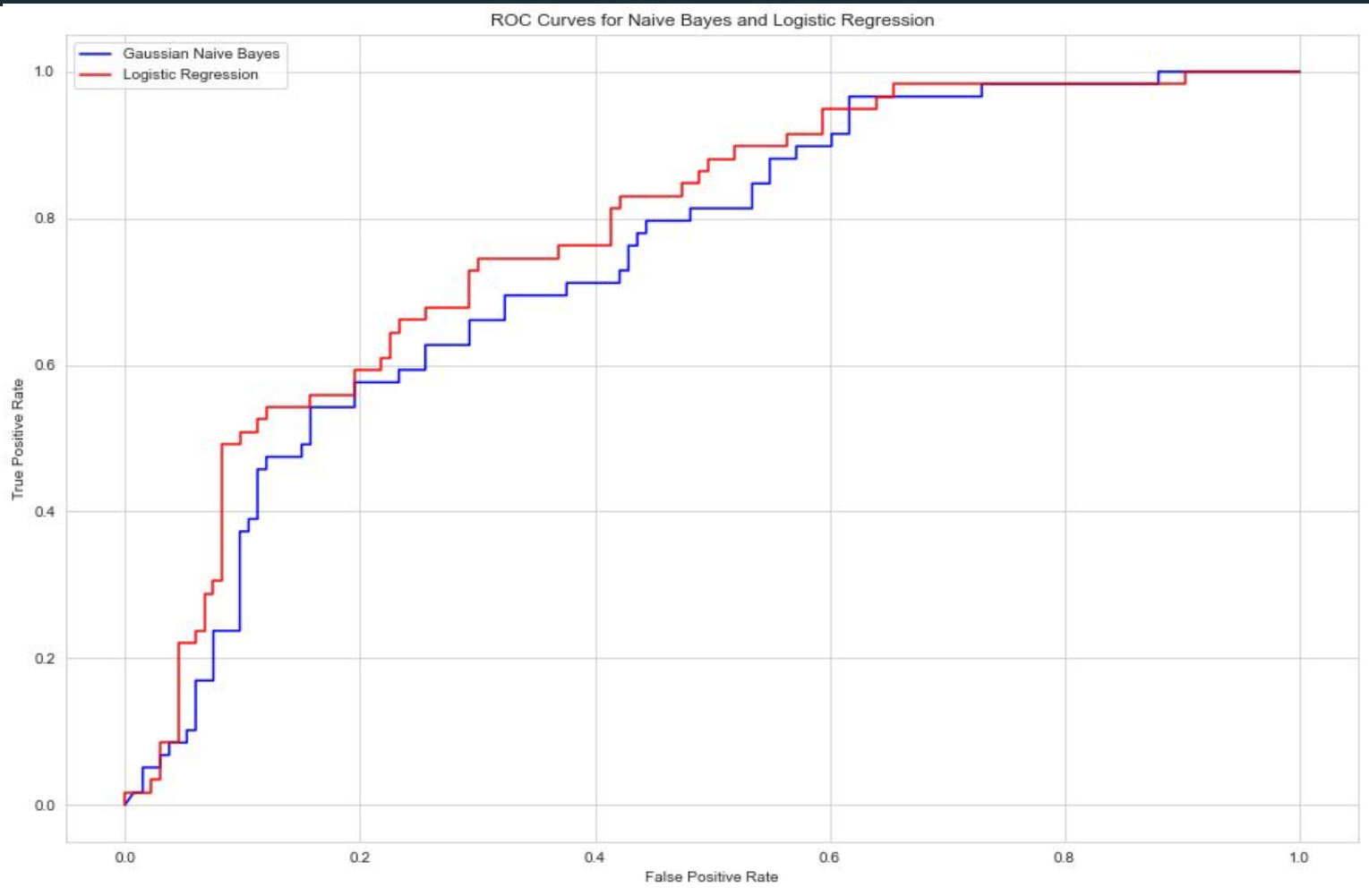
```
Gaussian Naive Bayes Report
===========================
Confusion Matrix
 [[105  25]
 [ 21  41]]
              precision    recall  f1-score   support

           0       0.83      0.81      0.82       130
           1       0.62      0.66      0.64        62

    accuracy                           0.76       192
   macro avg       0.73      0.73      0.73       192
weighted avg       0.76      0.76      0.76       192
```



# Naïve Bayes vs. Logistic Regression Model

- Logistic Regression wins the head-to-head
  - Better in all three metrics examined

```
Model Comparison
================
Out-of-Sample Accuracy: GaussianNB - 0.714; Logistic Regression - 0.755
Matthews Correlation Coefficient: GaussianNB - 0.350; Logistic Regression - 0.412
Out-of-Sample AUC: GaussianNB - 0.746; Logistic Regression - 0.781
```


ROC Curves for Naive Bayes and Logistic Regression
Gaussian Naive Bayes
Logistic Regression
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0

# Ideas to Increase Model Performance

- No real hyperparameters to tune
- Feature Selection
  - Marked Improvement
- Dimensionality Reduction
- Transform Data

```
Out-of-Sample Performance
=========================
Accuracy: 0.776
Matthews Correlation: 0.467
AUC: 0.782
```

Thank you Olivier, Nolan, Logan, & Erick for the resources!