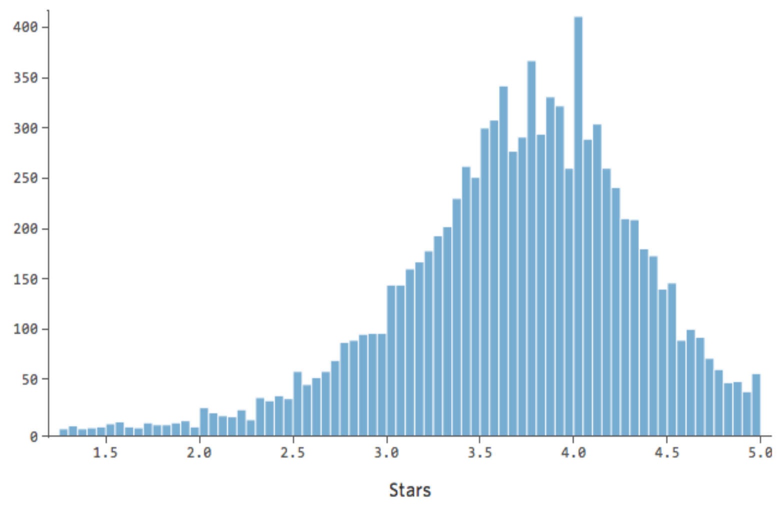
Probabilistic Classification

MACHINE LEARNING UNIT 12

Predict user’s ratings on a product

- Predict the ratings by a particular user to a given restaurant!
- Possible indicators:
  - The user’s ratings of other restaurants
    - most frequently, (s)he rates 4.0!



Predict user’s ratings on a product

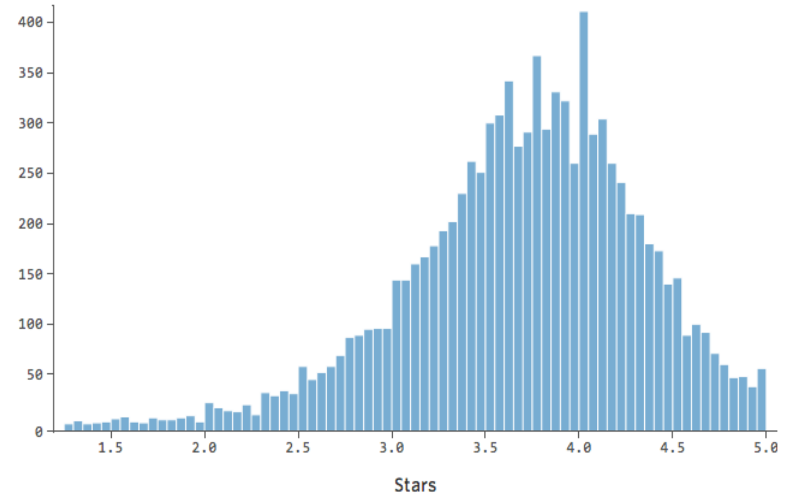
- Predict the ratings by a particular user to a given restaurant!
- Possible indicators:

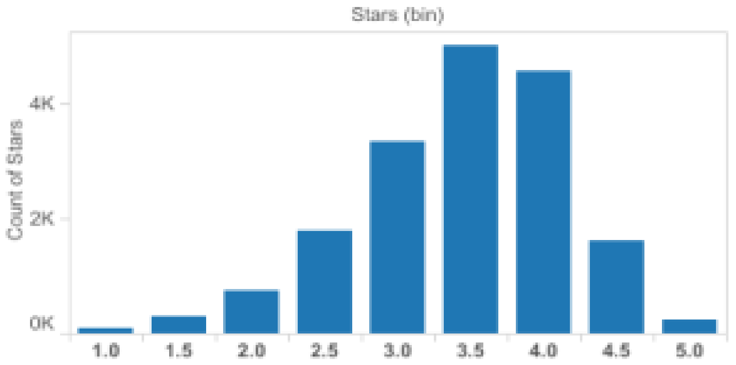
  - The user’s ratings of other restaurants

    - most frequently, (s)he rates 4.0!

  - Other users’ ratings of that restaurant

    - most frequently, others rate 3.5!





Predict user’s ratings on a product

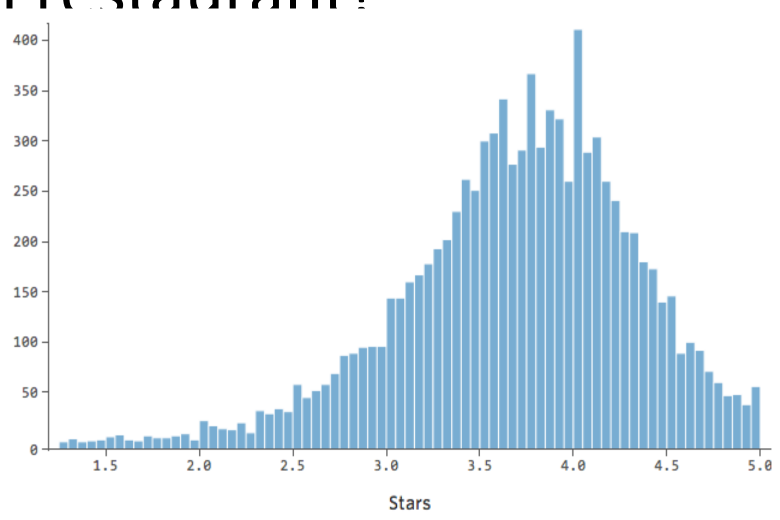
- Predict the ratings by a particular user to a given restaurant!
- Possible indicators:

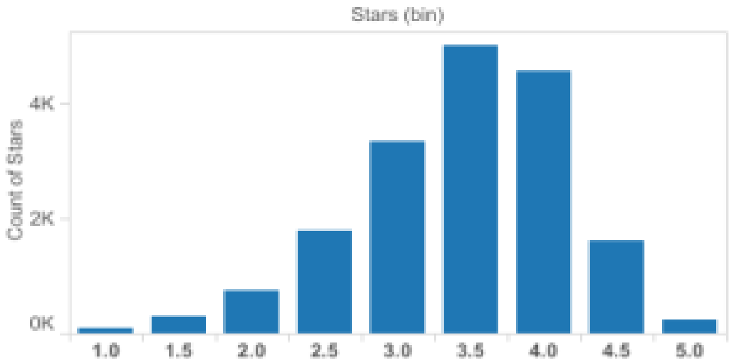
  - The user’s ratings of other restaurants

    - most frequently, (s)he rates 4.0!

  - Other users’ ratings of that restaurant

    - most frequently, others rate 3.5!

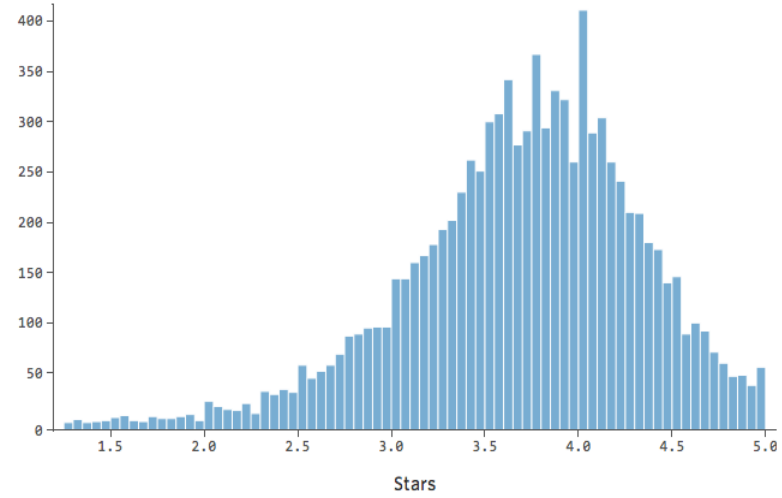


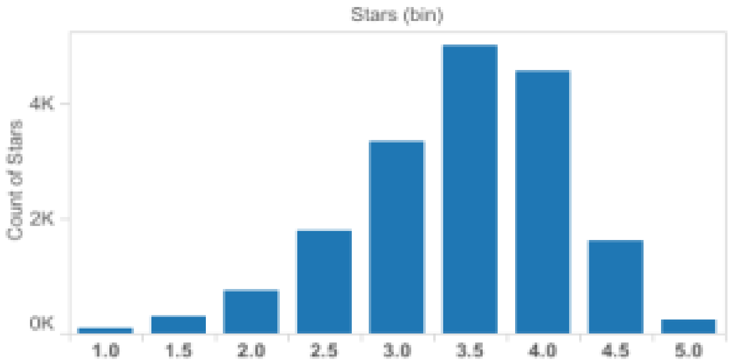


Predict user's ratings on a product

- Predict the ratings by a particular user to a given restaurant!
- Possible indicators:
  - The user's ratings of other restaurants
    - most frequently, (s)he rates 4.0!

  - Other users' ratings of that restaurant
    - most frequently, others rate 3.5!





- But we can never be sure!
- Can we attach a probability to each possible rating?

Generative Model

- A story about how the observed data were "born"
- Story in the language of probability!

- Treat labels Y and features X as random variables
- Story outline:

  - Y created first

  - X created based on Y

Generative Model

- A story about how the observed data were "born"
- Story in the language of probability!

- Treat labels Y and features X as random variables
- Story outline:
- - Y created first p(Y): prior distribution
- - X created based on Y p(X|Y):class-conditional distribution
- Probabilistic Classification: p(Y|X): posterior distribution

Prior Distribution

- Prior: information before observing X

	Y =1	Y = 2	Y = 3	Y = 4
	80	50	40	30

- P(Y = k) = ?
- Frequentist approach: just relative frequencies!

	Y = 1	Y = 2	Y = 3	Y = 4
	0.4	0.25	0.20	0.15

Posterior Distribution

- Posterior: information after observing X
- $P(Y = k \mid X) = ?$

- Bayes Theorem:
- $P(Y = k \mid X) = (p(X \mid Y = k) * p(Y = k)) / p(X)$

$$= K * p(X \mid Y = k) * p(Y = k)$$

Class-conditional Distribution

	Y = 1	Y = 2	Y = 3	Y = 4
X < 15	40	45	10	5
X > 15	40	5	30	25

- $P(X \mid Y = k) = ??$
- $P(X<15 \mid Y=1) = 40/(40+40) = 0.5$
- $P(X>15 \mid Y=3) = 30/(30+10) = 0.75$

- $P(Y = k \mid X) = ???$

Frequentist Approach: Direct estimation

	Y = 1	Y = 2	Y = 3	Y = 4	
X < 15	40	45	10	5	100
X > 15	40	5	30	25	100

- $P(Y = 1 \mid X<15) = 40 / 100 = 0.4$

	Y = 1	Y = 2	Y = 3	Y = 4	
$p(Y \mid X < 15)$	0.4	0.45	0.10	0.05	1.0
$p(Y \mid X > 15)$	0.4	0.05	0.30	0.25	1.0

- Similar to Decision Trees

Bayesian Approach: Posterior Distribution

	Y = 1	Y = 2	Y = 3	Y = 4
X < 15	40	45	10	5
X > 15	40	5	30	25

- P(Y = 1 | X<15) = K * p(X<15 | Y=1) * p(Y=1) = K*(40/80)*(80/200)
- P(Y = 2 | X<15) = K * p(X<15 | Y=2) * p(Y=2) = K*(45/50)*(50/200)
- P(Y = 3 | X<15) = K * p(X<15 | Y=3) * p(Y=3) = K*(10/40)*(40/200)
- P(Y = 4 | X<15) = K * p(X<15 | Y=4) * p(Y=4) = K* (5/30)*(30/200)
- K = ???

Posterior Distribution

	Y = 1	Y = 2	Y = 3	Y = 4
X < 15	40	45	10	5
X > 15	40	5	30	25

	Y = 1	Y = 2	Y = 3	Y = 4
Prior p(Y)	0.40	0.25	0.20	0.15
$p(Y \mid X < 15)$	0.40	0.45	0.10	0.05
$P(Y \mid X > 15)$	0.40	0.05	0.30	0.25

Probabilistic Classifier

- Predicted label : mode of the posterior distribution!
- $Y_{pred} = \text{argmax}_k\ p\ (Y = k \mid X)$
- Confidence of the prediction = $p(Y = Y_{pred} \mid X)$

- If Bayesian approach used for $p(Y \mid X)$: Bayesian Classifier!

Posterior Distribution

	Y = 1	Y = 2	Y = 3	Y = 4
X < 15	40	45	10	5
X > 15	40	5	30	25

	Y = 1	Y = 2	Y = 3	Y = 4	Ypred
Prior	0.40	0.25	0.20	0.15	1
X < 15	0.40	0.45	0.10	0.05	2
X > 15	0.40	0.05	0.30	0.25	1

Multi-dimensional Features

	Y = 1	Y = 2	Y = 3	Y = 4
X1<15	40	45	10	5
X1>15	40	5	30	25
X2 = a	40	30	15	30
X2 = b	40	20	25	0

- $P(Y = k \mid X1=12, X2=a) = ????$
- We need Joint Distribution of the features!!!

Multi-dimensional Features

	Y = 1	Y = 2	Y = 3	Y = 4
X1<15, X2=a	30	25	0	5
X1<15, X2=b	10	20	10	0
X1>15, X2=a	10	5	15	25
X1>15, X2=b	30	0	15	0

- $P(Y = 3 \mid X1=12, X2 =b) = K * P(X1<15, X2=b \mid Y = 3) *P(Y=3)$

  $= ?$

Multi-dimensional Features

	Y = 1	Y = 2	Y = 3	Y = 4
X1<15, X2=a	0.375	0.50	0	0.166
X1<15, X2=b	0.125	0.40	0.25	0
X1>15, X2=a	0.125	0.10	0.375	0.837
X1>15, X2=b	0.375	0	0.375	0

- $P(Y = 3 \mid X1=12, X2 =b) = K * P(X1<15, X2=b \mid Y = 3) *P(Y=3)$

$$= K * 0.25 * 0.20$$

Multi-dimensional Features

	Y = 1	Y = 2	Y = 3	Y = 4
X1<15, X2=a	0.375 * 0.4 * K1	0.50 * 0.25 * K1	0 * 0.2 * K1	0.166 * 0.15 * K1
X1<15, X2=b	0.125 * 0.4 * K2	0.40 * 0.25 * K2	0.25 * 0.2 * K2	0 * 0.15 * K2
X1>15, X2=a	0.125 * 0.4 * K3	0.10 * 0.25 * K3	0.375 * 0.2 * K3	0.837 * 0.15 * K3
X1>15, X2=b	0.375 * 0.4 * K4	0 * 0.25 * K4	0.375 * 0.2 * K4	0 * 0.15 * K4

- $P(Y = 3 \mid X1=12, X2 = b) = K2 * P(X1<15, X2=b \mid Y = 3) * P(Y=3)$

$$= K2 * 0.25 * 0.20$$

$$= ??$$

Naïve Bayes Classifier

- D-dimensional feature vector, M values each
- Rows of table = M**D

- Assumption: all features are independent (Naïve!)
- $P(X1<15, X2 =b) = p(X1<15) * p(X2=b)$

- D tables, rows of each table = M
- Naïve, but computationally efficient!

Naïve Bayes Classification

- $P(Y = k \mid X1<15, X2=b) = K * p(X1<15, X2=b \mid Y = k) * p(Y = k)$
  $= K * p(X1<15 \mid Y = k) * p(X2=b \mid Y = k) * p(Y = k)$

Final prediction $= \text{argmax}_k\ p(X_1|Y=k)\ p(X_2|Y=k)*\ldots\ldots\ p(X_D|Y=k)*p(Y=k)$

Confidence $= \max_k\ p(X_1|Y=k)\ p(X_2|Y=k)*\ldots\ldots\ p(X_D|Y=k)*p(Y=k)$

```
import numpy as np
```

```
def fit(X_train,Y_train):
    result ={}
    class_values = set(Y_train)
    for current_class in class_values:
        result[current_class] = {}
        result["total_data"] = len(Y_train)
        current_class_rows = (Y_train == current_class)
        X_train_current = X_train[current_class_rows]
        Y_train_current = Y_train[current_class_rows]
        num_features = X_train.shape[1]
        result[current_class]["total_count"] = len(Y_train_current)
        for j in range(1,num_features+1):
            result[current_class][j] ={}
            all_possible_values = set(X_train[:,j-1])
            for current_value in all_possible_values:
                result[current_class][j][current_value] = (X_train_current[:,j-1] == current_value).sum()
    return result
```

```
def probablity(dictionary,x,current_class):
    output= np.log(dictionary[current_class]["total_count"])-np.log(dictionary["total_data"])
    num_features = len(dictionary[current_class].keys())-1;
    for j in range(1,num_features+1):
        xj = x[j-1]
        count_current_class_with_value_xj = dictionary[current_class][j][xj] + 1
        count_current_class = dictionary[current_class]["total_count"] + len(dictionary[current_class][j].keys())
        current_xj_prob = np.log(count_current_class_with_value_xj) -np.log(count_current_class)
        output = output + current_xj_prob
    return output
```

```
def predictSinglePoint(dictionary,x):
    classes = dictionary.keys()
    best_p  = -1000
    best_class = -1
    first_run = True
    for current_class in classes:
        if(current_class == "total_data"):
            continue
        p_current_class = probablity(dictionary,x,current_class)
        if(first_run or p_current_class > best_p):
            best_p = p_current_class
            best_class = current_class
        first_run = False
    return best_class
```

```
def predict(dictonary, X_test):
    y_pred = []
    for x in X_test:
        x_class = predictSinglePoint(dictionary,x)
        y_pred.append(x_class)
    return y_pred
```