UTKARSH CHAUBE 202214195

HOME ASSIGNMENT 7

a. Prepare the dictionary which counts the number of “yes” and “no” class labels for each nominal value in an attribute.

```
import pandas as pd

def count_class_labels(df, attribute):
    # Prepare a dictionary that counts the number of "yes" and "no" class
labels for each nominal value in an attribute
    counts_dict = {}
    unique_values = df[attribute].unique()

    for value in unique_values:
        counts_dict[value] = {
            'yes': df[(df[attribute] == value) & (df['buys_computer'] ==
'yes')].shape[0],
            'no': df[(df[attribute] == value) & (df['buys_computer'] ==
'no')].shape[0]
        }

    return counts_dict

# Load the dataset
df = pd.read_csv('buys_computer.csv')

# Example usage:
counts_dict = count_class_labels(df, 'age')
print(counts_dict)
```

b.Compute the probability of the class labels (buys_computer=”yes” or buys_computer= ”no”) for each nominal value in an attribute.

```
def compute_class_label_probabilities(df, attribute):
    # Compute the probability of the class labels for each nominal value
```

```
in an attribute
    class_label_probs = {}
    unique_values = df[attribute].unique()
    total_samples = len(df)

    for value in unique_values:
        value_df = df[df[attribute] == value]
        yes_prob = value_df[value_df['buys_computer'] == 'yes'].shape[0] /
total_samples
        no_prob = value_df[value_df['buys_computer'] == 'no'].shape[0] /
total_samples
        class_label_probs[value] = {'yes': yes_prob, 'no': no_prob}

    return class_label_probs

# Example usage:
class_label_probs = compute_class_label_probabilities(df, 'age')
print(class_label_probs)
```

C. Using Bayes' theorem compute the probability of instance X = (age = youth, income = medium, student = yes, credit_rating = fair) against class label (buys_computer="yes" and buys_computer="no").

```
def compute_bayes_probability(df, instance_values, class_label):
    # Compute the probability of instance X against class label
(buys_computer="yes" and buys_computer="no") using Bayes' theorem
    total_samples = len(df)
    class_counts = df[class_label].value_counts()
    class_probs = class_counts / total_samples

    bayes_probs = {}
    for label in class_probs.index:
        prob = class_probs[label]
        for attr, value in instance_values.items():
            subset = df[df[attr] == value]
            subset_count = subset[class_label].value_counts().get(label,
0)
            prob *= (subset_count + 1) / (class_counts[label] +
len(subset[attr].unique()))
```

```
        bayes_probs[label] = prob

    return bayes_probs

# Define the instance X
instance_values = {'age': 'youth', 'income': 'medium', 'student': 'yes',
'credit_rating': 'fair'}

# Example usage:
bayes_probs = compute_bayes_probability(df, instance_values,
'buys_computer')
print(bayes_probs)
```

D. Justify the probability of X belongs to a particular class label (buys_computer=”yes” or buys_computer=”no”).

```
def justify_bayes_probability(bayes_probs):
    # Justify the probability of X belonging to a particular class label
(buys_computer="yes" or buys_computer="no")
    max_prob_label = max(bayes_probs, key=bayes_probs.get)
    justification = f"The instance is more likely to belong to class
'{max_prob_label}' with a probability of
{bayes_probs[max_prob_label]:.4f}"
    return justification

# Example usage:
justification = justify_bayes_probability(bayes_probs)
print(justification)
```