

LAMPIRAN A

Lampiran ini berisi coding main class yang diberi nama ClassLatihan.java. Bagian ini adalah bagian utama. Ini adalah kelas utama yang menjalankan program pada penelitian ini.

```
package packageLatihan;

import com.mycompany.mavenproject1.NgramTest;
import java.util.Iterator;
import java.util.List;
import java.util.logging.Level;
import java.util.logging.Logger;
import model.Probabilitas;
import twitter4j.Query;
import twitter4j.QueryResult;
import twitter4j.Status;
import twitter4j.Twitter;
import twitter4j.TwitterException;
import twitter4j.TwitterFactory;
import twitter4j.conf.ConfigurationBuilder;

/**
 *
 * @author alzygumgum
 */
public class ClassLatihan {

    public static void main(String[] args) {
        try {
            ConfigurationBuilder cb = new ConfigurationBuilder();
            cb.setDebugEnabled(true)
```

```

        .setOAuthConsumerKey("YS3r38C3XpmO4LmK4mZRD5UD0")

        .setOAuthConsumerSecret("8CTkq8x62AKUu327Y0UchE8EqGVrybQcTWcWdUYEogdb
        AmYxby")

        .setOAuthAccessToken("65427688-
        ezABWUv1F9i3WHxS25hIDDD3o6FIRmY4bMhX9nl54")

        .setOAuthAccessTokenSecret("soxsVaJzR413JPJ67lTswmfWjEm5LTOavRTuDFsnLD3i")
    ;

```

```

    TwitterFactory tf = new TwitterFactory(cb.build());
    Twitter twitter = tf.getInstance();
    Query query = new Query("@basuki_btp");
    QueryResult result = twitter.search(query);
    ProsesClass proses = new ProsesClass();

    for (Status status : result.getTweets()) {
        String line;
        line = status.getText();
        String pure = line.replaceAll("@[a-zA-Z0-9_]+", "");
        pure = pure.replaceAll("#(\\w+)", "");
        pure = pure.replaceAll("\\!(\\p{L}?)", "");
        pure = pure.replaceAll("http[^\s]+", "");
        pure = pure.toLowerCase();
        pure = pure.replaceAll("\\brt()?\b", "");
        pure = pure.replaceAll("\\W\\s+", " ");
        List stpwrdrResult = proses.stopWord();
        Iterator itstop = stpwrdrResult.iterator();
        while (itstop.hasNext()) {
            pure = pure.replaceAll((String) itstop.next(), " ");
        }
        pure = pure.trim();
        List ngramResult = NgramTest.ngrams(1, pure);
    }

```

```

        System.out.println(line);
        System.out.println(pure);
        int id_tweet = proses.insertTweet(status.getUser().getScreenName(), pure);
        Iterator it = ngramResult.iterator();
        if (id_tweet != -1) {
            while (it.hasNext()) {
                proses.insertNgrams(id_tweet, (String) it.next());
            }
        }
        proses.pindahNgram();
        List<Probabilitas> t = proses.selectProbabilitas();
        proses.updateSentimen(t);
        List<Integer> i = proses.getTweet();
        proses.hitungProbabilitas(i);
        proses.intisari();
        proses.updateTweetMaster();
    status.getText());
    }
    } catch (TwitterException ex) {
        Logger.getLogger(ClassLatihan.class.getName()).log(Level.SEVERE, null, ex);
    }
    }
}

```

LAMPIRAN B

Lampiran ini berisi class yang membuat koneksi antara java dan database MySQL. Class ini diberi nama “ConnectDatabaseClass.java”.

```
package packageLatihan;

import static com.mycompany.mavenproject1.NgramTest.concat;
import com.mysql.jdbc.Statement;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.HashSet;
import java.util.Iterator;
import java.util.List;
import java.util.logging.Level;
import java.util.logging.Logger;
import model.Probabilitas;

/**
 *
 * @author alzygumgum
 */
public class ConnectDatabaseClass {

    Connection connections;

    public void closeConnection() {
        try {
            connections.close();
        }
    }
}
```

```

        connections = null;
    } catch (SQLException ex) {
        Logger.getLogger(ConnectDatabaseClass.class.getName()).log(Level.SEVERE, null,
ex);
    }
}

```

```

public Connection getConnection() {
    if(connections == null){
        createConnection();
    }
    return connections;
}

```

```

public void createConnection() {
    String dbUrl =
"jdbc:mysql://localhost/anasentimen?useUnicode=true&characterEncoding=UTF-8";
    String dbClass = "com.mysql.jdbc.Driver";
    String query = "Select distinct(word) from INFORMATION_SCHEMA.TABLES";
    String username = "root";
    String password = "sayagaktau";
    try {
        Class.forName(dbClass);
        if(connections == null){
            connections = DriverManager.getConnection(dbUrl,
                username, password);
        }
    } catch (Exception e) {
    }
}
}

```

LAMPIRAN C

Lampiran ini berisi class yang mengeksekusi dokumen tweet dengan fitur n-gram. Class ini memisahkan kalimat menjadi kata-per-kata. Class ini diberi nama “NgramTest.java”.

```
package com.mycompany.mavenproject1;

import java.util.ArrayList;
import java.util.List;
/**
 *
 * @author alzygungum
 */
public class NgramTest {
    public static List<String> ngrams(int n, String str) {
        List<String> ngrams = new ArrayList<>();
        String[] words = str.split(" ");
        for (int i = 0; i < words.length - n + 1; i++)
            ngrams.add(concat(words, i, i+n));
        return ngrams;
    }

    public static String concat(String[] words, int start, int end) {
        StringBuilder sb = new StringBuilder();
        for (int i = start; i < end; i++)
            sb.append(i > start ? " " : "").append(words[i]);
        return sb.toString();
    }
}
```

LAMPIRAN D

Lampiran ini berisi class yang berisi proses *text preprocessing* sampai kepada perhitungan Algoritma Naive Bayes Classifier. Class ini diberi nama “ProsesClass.java”.

```
package packageLatihan;

import com.mysql.jdbc.Statement;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.Iterator;
import java.util.List;
import java.util.logging.Level;
import java.util.logging.Logger;
import model.Probabilitas;

/**
 *
 * @author alzygumgum
 */
public class ProsesClass {

    public int insertTweet(String user, String tweet) {
        try {
            ConnectDatabaseClass db = new ConnectDatabaseClass();
            Connection connections = db.getConnection();
            PreparedStatement preparedStatement
```

```

        = connections.prepareStatement("INSERT into tweetmaster (user, tweet)
VALUES (?,?)",
        Statement.RETURN_GENERATED_KEYS);
preparedStatement.setString(1, user);
preparedStatement.setString(2, tweet);

preparedStatement.executeUpdate();

ResultSet tableKeys = preparedStatement.getGeneratedKeys();
tableKeys.next();
int autoGeneratedID = tableKeys.getInt(1);
db.closeConnection();
return autoGeneratedID;
} catch (SQLException ex) {
    Logger.getLogger(ConnectDatabaseClass.class.getName()).log(Level.SEVERE, null,
ex);
}
return -1;//error
}

```

```

public int insertNgrams(int id_tweet, String tweet) {
    try {
        ConnectDatabaseClass db = new ConnectDatabaseClass();
        Connection connections = db.getConnection();
        PreparedStatement preparedStatement
            = connections.prepareStatement("INSERT into tweetngram (id_tweet, ngram)
VALUES (?,?)",
            Statement.RETURN_GENERATED_KEYS);
        preparedStatement.setInt(1, id_tweet);
        preparedStatement.setString(2, tweet);
    }
}

```



```

preparedStatement.executeUpdate();

ResultSet tableKeys = preparedStatement.getGeneratedKeys();
tableKeys.next();
int autoGeneratedID = tableKeys.getInt(1);
db.closeConnection();
return autoGeneratedID;
} catch (SQLException ex) {
    Logger.getLogger(ConnectDatabaseClass.class.getName()).log(Level.SEVERE, null,
ex);
}
return -1;//error
}

```

```

public void deleteData() {
    try {
        ConnectDatabaseClass db = new ConnectDatabaseClass();
        Connection connections = db.getConnection();
        String SQL = "DELETE FROM probabilitas";
        PreparedStatement pstmt1 = null;
        pstmt1 = connections.prepareStatement(SQL);
        pstmt1.executeUpdate();

        SQL = "DELETE FROM tweetngram";
        PreparedStatement pstmt = null;
        pstmt = connections.prepareStatement(SQL);
        pstmt.executeUpdate();

        SQL = "DELETE FROM tweetmaster";
        pstmt = null;
        pstmt = connections.prepareStatement(SQL);
    }
}

```

```

        pstmt.executeUpdate();
        db.closeConnection();
    } catch (SQLException ex) {
        Logger.getLogger(ConnectDatabaseClass.class.getName()).log(Level.SEVERE, null,
ex);
    }
}

```

```

public List<String> stopWord() {
    List<String> stpwrđ = new ArrayList<>();
    try {
        ConnectDatabaseClass db = new ConnectDatabaseClass();
        Connection connections = db.getConnection();
        String SQL = "SELECT stopword FROM stoplist";
        PreparedStatement pstmt = null;
        pstmt = connections.prepareStatement(SQL);
        ResultSet rs = pstmt.executeQuery();
        while (rs.next()) {
            String stopword = rs.getString("stopword");
            stpwrđ.add(stopword);
        }
        db.closeConnection();
    } catch (SQLException ex) {
        Logger.getLogger(ConnectDatabaseClass.class.getName()).log(Level.SEVERE, null,
ex);
    }
    return stpwrđ;
}

```

```

public List<Integer> getTweet() {

```

```

List<Integer> tweet = new ArrayList<>();
try {
    ConnectDatabaseClass db = new ConnectDatabaseClass();
    Connection connections = db.getConnection();
    String SQL = "SELECT * FROM tweetmaster where sentimen is NULL";
    PreparedStatement pstmt = null;
    pstmt = connections.prepareStatement(SQL);
    ResultSet rs = pstmt.executeQuery();
    while (rs.next()) {
        int id_tweet = rs.getInt("id_tweet");
        Integer t = new Integer(id_tweet);
        tweet.add(t);
    }
    db.closeConnection();
} catch (Exception e) {
    Logger.getLogger(ConnectDatabaseClass.class.getName()).log(Level.SEVERE, null,
e);
}
return tweet;
}

public void pindahNgram() {
    try {
        ConnectDatabaseClass db = new ConnectDatabaseClass();
        Connection connections = db.getConnection();
        String SQL = "SELECT * FROM tweetmaster where sentimen is NULL";
        PreparedStatement pstmt = null;
        pstmt = connections.prepareStatement(SQL);
        ResultSet rs = pstmt.executeQuery();
        while (rs.next()) {
            int id_tweet = rs.getInt("id_tweet");
            SQL = "SELECT * FROM tweetngram where id_tweet = ?";

```

```

PreparedStatement pstmt2 = null;
ConnectDatabaseClass db1 = new ConnectDatabaseClass();
Connection con = db1.getConnection();
pstmt2 = con.prepareStatement(SQL);
pstmt2.setInt(1, id_tweet);
ResultSet rs2 = pstmt2.executeQuery();
while (rs2.next()) {
    int id_ngram = rs2.getInt("id_ngram");
    String ngram = rs2.getString("ngram");
    SQL = "SELECT * FROM probabilitas a join tweetngram b on(a.id_ngram =
b.id_ngram) join tweetmaster c on(b.id_tweet = c.id_tweet) where c.id_tweet = ? and a.ngram
like ?";

```

```

PreparedStatement pstmt3 = null;
ConnectDatabaseClass db2 = new ConnectDatabaseClass();
Connection con1 = db2.getConnection();
pstmt3 = con1.prepareStatement(SQL);
pstmt3.setInt(1, id_tweet);
pstmt3.setString(2, ngram);
ResultSet rs3 = pstmt3.executeQuery();
int id_prob = -1;
int qty = 1;
while (rs3.next()) {
    id_prob = rs3.getInt("id_prob");
    qty = rs3.getInt("qty");
    SQL = "UPDATE probabilitas set qty = ? where id_prob = ?";
    PreparedStatement pstmt5 = null;
    ConnectDatabaseClass db3 = new ConnectDatabaseClass();
    Connection con2 = db3.getConnection();
    pstmt5 = con2.prepareStatement(SQL);
    qty = qty + 1;
    pstmt5.setInt(1, qty);

```

```

        pstmt5.setInt(2, id_prob);
        pstmt5.executeUpdate();
        db3.closeConnection();
    }
    if (id_prob == -1) {
        SQL = "INSERT INTO probabilitas (id_ngram, ngram, qty) VALUES (?, ?,
?);";

        PreparedStatement pstmt4 = null;
        ConnectDatabaseClass db4 = new ConnectDatabaseClass();
        Connection con3 = db4.getConnection();
        pstmt4 = con3.prepareStatement(SQL);
        pstmt4.setInt(1, id_ngram);
        pstmt4.setString(2, ngram);
        pstmt4.setInt(3, qty);
        pstmt4.executeUpdate();
        db4.closeConnection();
    }
    db2.closeConnection();
}
db1.closeConnection();
}
db.closeConnection();
} catch (SQLException ex) {
    Logger.getLogger(ConnectDatabaseClass.class.getName()).log(Level.SEVERE, null,
ex);
}
}

public void hitungProbabilitas(List<Integer> idtweet) {
    try {
        ConnectDatabaseClass db = new ConnectDatabaseClass();
        Connection connections = db.getConnection();

```

```

Iterator it = idtweet.iterator();

while (it.hasNext()) {
    String SQL = "SELECT * FROM probabilitas a join tweetngram b on(a.id_ngram
= b.id_ngram) join tweetmaster c on(b.id_tweet = c.id_tweet) where c.id_tweet=?";

    PreparedStatement pstmt = null;

    pstmt = connections.prepareStatement(SQL);

    Integer angka = (Integer) it.next();

    pstmt.setInt(1, (int) angka.intValue());

    ResultSet rs = pstmt.executeQuery();

    while (rs.next()) {
        int id_tweet = rs.getInt("id_tweet");

        int id_ngram = rs.getInt("id_ngram");

        int id_prob = rs.getInt("id_prob");

        String sentimen = rs.getString("sentimen");

        int nk = rs.getInt("qty");

        SQL = "SELECT SUM(qty)as n FROM probabilitas a JOIN tweetngram b
ON(a.id_ngram = b.id_ngram) JOIN tweetmaster c ON(b.id_tweet = c.id_tweet) WHERE
c.id_tweet = ? AND a.sentimen = ?";

        PreparedStatement pstmt2 = null;

        ConnectDatabaseClass db1 = new ConnectDatabaseClass();

        Connection con1 = db1.getConnection();

        pstmt2 = con1.prepareStatement(SQL);

        pstmt2.setInt(1, id_tweet);

        pstmt2.setString(2, sentimen);

        ResultSet rs2 = pstmt2.executeQuery();

        int n = 0;

        while (rs2.next()) {
            n = rs2.getInt("n");
        }
    }
}

```

```
db1.closeConnection();
```

```
SQL = "SELECT SUM(qty) as vocab FROM probabilitas a JOIN tweetngram b  
ON(a.id_ngram = b.id_ngram) JOIN tweetmaster c ON(b.id_tweet = c.id_tweet) WHERE  
c.id_tweet = ? ";
```

```
PreparedStatement pstmt3 = null;
```

```
ConnectDatabaseClass db2 = new ConnectDatabaseClass();
```

```
Connection con2 = db2.getConnection();
```

```
pstmt3 = con2.prepareStatement(SQL);
```

```
pstmt3.setInt(1, id_tweet);
```

```
ResultSet rs3 = pstmt3.executeQuery();
```

```
int vocab = 0;
```

```
while (rs3.next()) {
```

```
    vocab = rs3.getInt("vocab");
```

```
}
```

```
db2.closeConnection();
```

```
float up = (nk + 1);
```

```
float down = (n + vocab);
```

```
float hasil = up / down;
```

```
SQL = "UPDATE probabilitas set prob = ? where id_prob = ? and id_ngram =  
?";
```

```
PreparedStatement pstmt4 = null;
```

```
ConnectDatabaseClass db3 = new ConnectDatabaseClass();
```

```
Connection con3 = db3.getConnection();
```

```
pstmt4 = con3.prepareStatement(SQL);
```

```
pstmt4.setFloat(1, hasil);
```

```
pstmt4.setInt(2, id_prob);
```

```
pstmt4.setInt(3, id_ngram);
```

```
pstmt4.executeUpdate();
```

```

        db3.closeConnection();
    }
}

b.id_ngram) join tweetmaster c on(b.id_tweet = c.id_tweet) "
    db.closeConnection();
} catch (SQLException ex) {
    Logger.getLogger(ConnectDatabaseClass.class.getName()).log(Level.SEVERE, null,
ex);
}
}
}

```

```

public List<Probabilitas> selectProbabilitas() {
    List<Probabilitas> selProbs = new ArrayList<>();
    try {
        ConnectDatabaseClass db = new ConnectDatabaseClass();
        Connection connections = db.getConnection();
        String SQL = "SELECT * FROM probabilitas where sentimen is NULL";
        PreparedStatement pstmt = null;
        pstmt = connections.prepareStatement(SQL);
        ResultSet rs = pstmt.executeQuery();
        while (rs.next()) {
            Probabilitas probs = new Probabilitas();
            int id_prob = rs.getInt("id_prob");
            int id_ngram = rs.getInt("id_ngram");
            String ngram = rs.getString("ngram");
            int qty = rs.getInt("qty");
            String sentimen = rs.getString("sentimen");
            float prob = rs.getFloat("prob");
            probs.setId_prob(id_prob);
            probs.setId_ngram(id_ngram);

```



```

        probs.setNgram(ngram);
        probs.setQty(qty);
        probs.setSentimen(sentimen);
        probs.setProb(prob);
        selProbs.add(probs);
    }
    db.closeConnection();
} catch (SQLException ex) {
    Logger.getLogger(ConnectDatabaseClass.class.getName()).log(Level.SEVERE, null,
ex);
}
return selProbs;
}

```

```

public void updateSentimen(List<Probabilitas> t) {

    try {
        ConnectDatabaseClass db = new ConnectDatabaseClass();
        Connection connections = db.getConnection();
        Iterator it = t.iterator();

        while (it.hasNext()) {

            String SQL = "SELECT * FROM sentimen where kata = ?";
            PreparedStatement pstmt = null;
            pstmt = connections.prepareStatement(SQL);
            Probabilitas o = (Probabilitas) it.next();
            pstmt.setString(1, o.getNgram());
            ResultSet rs = pstmt.executeQuery();
            String sentimen = "netral";
            while (rs.next()) {

```

```

        sentimen = rs.getString("sentimen");
    }
    SQL = "UPDATE probabilitas set sentimen = ? where id_ngram = ? and id_prob =
?";

    pstmt = null;
    pstmt = connections.prepareStatement(SQL);
    pstmt.setString(1, sentimen);
    pstmt.setInt(2, o.getId_ngram());
    pstmt.setInt(3, o.getId_prob());
    pstmt.executeUpdate();
}
db.closeConnection();
} catch (SQLException ex) {
    Logger.getLogger(ConnectDatabaseClass.class.getName()).log(Level.SEVERE, null,
ex);
}
}

```

```

public void intisari() {
    try {
        ConnectDatabaseClass db = new ConnectDatabaseClass();
        Connection connections = db.getConnection();

        String SQL = "SELECT c.id_tweet,a.sentimen,SUM(prob) as total_prob FROM
probabilitas a JOIN tweetngram b ON (a.id_ngram = b.id_ngram) JOIN tweetmaster c ON
(b.id_tweet = c.id_tweet) GROUP BY c.id_tweet, a.sentimen";

        PreparedStatement pstmt = null;
        pstmt = connections.prepareStatement(SQL);
        ResultSet rs = pstmt.executeQuery();
        while (rs.next()) {

            int id_tweet = rs.getInt("id_tweet");
            String sentimen = rs.getString("sentimen");

```

```

float prob = rs.getFloat("total_prob");

SQL = "INSERT INTO intisari (id_tweet, sentimen, prob) VALUES (?, ?, ?)";
ConnectDatabaseClass db2 = new ConnectDatabaseClass();
Connection con = db2.getConnection();
pstmt = con.prepareStatement(SQL);
pstmt.setInt(1, id_tweet);
pstmt.setString(2, sentimen);
pstmt.setFloat(3, prob);
pstmt.executeUpdate();
db2.closeConnection();
}
db.closeConnection();
} catch (SQLException ex) {
    Logger.getLogger(ConnectDatabaseClass.class.getName()).log(Level.SEVERE, null,
ex);
}

}

public void updateTweetMaster() {
    try {
        ConnectDatabaseClass db = new ConnectDatabaseClass();
        Connection connections = db.getConnection();

        where a.prob = (select max(b.prob) as max_prob from intisari b where b.sentimen <> 'netral'
and b.id_tweet=a.id_tweet group by b.id_tweet limit 1)";

        String SQL = "SELECT a.id_tweet, a.id_intisari, a.sentimen,a.prob FROM intisari a
\n"
        + "INNER JOIN (SELECT id_tweet,MAX(b.prob) AS prob FROM intisari b
\n"
        + "WHERE b.sentimen <> 'netral' \n"

```

```
        + "GROUP BY b.id_tweet) ss ON a.id_tweet = ss.id_tweet AND a.prob =  
ss.prob";
```

```
PreparedStatement pstmt = null;
```

```
pstmt = connections.prepareStatement(SQL);
```

```
ResultSet rs = pstmt.executeQuery();
```

```
while (rs.next()) {
```

```
    //int id_intisari = rs.getInt("id_intisari");
```

```
    int id_tweet = rs.getInt("id_tweet");
```

```
    String sentimen = rs.getString("sentimen");
```

```
    float prob = rs.getFloat("prob");
```

```
    SQL = "UPDATE tweetmaster set sentimen = ? where id_tweet = ?";
```

```
    ConnectDatabaseClass db1 = new ConnectDatabaseClass();
```

```
    Connection con = db1.getConnection();
```

```
    pstmt = con.prepareStatement(SQL);
```

```
    pstmt.setString(1, sentimen);
```

```
    pstmt.setInt(2, id_tweet);
```

```
    pstmt.executeUpdate();
```

```
    db1.closeConnection();
```

```
}
```

```
SQL = "UPDATE tweetmaster set sentimen = ? where sentimen is NULL";
```

```
ConnectDatabaseClass db1 = new ConnectDatabaseClass();
```

```
Connection con = db1.getConnection();
```

```
pstmt = con.prepareStatement(SQL);
```

```
pstmt.setString(1, "netral");
```

```
//pstmt.setInt(2, id_tweet);
```

```
pstmt.executeUpdate();
```

```
db1.closeConnection();
```

```
} catch (SQLException ex) {
```

```

        Logger.getLogger(ConnectDatabaseClass.class.getName()).log(Level.SEVERE, null,
ex);
    }
}

```

```

public void intiToMaster() {
    try {
        ConnectDatabaseClass db = new ConnectDatabaseClass();
        Connection connections = db.getConnection();

        String SQL = "SELECT c.id_tweet,a.sentimen,SUM(prob) FROM probabilitas a
JOIN tweetngram b ON (a.id_ngram = b.id_ngram) JOIN tweetmaster c ON (b.id_tweet =
c.id_tweet) GROUP BY c.id_tweet, a.sentimen";

        PreparedStatement pstmt = null;
        pstmt = connections.prepareStatement(SQL);
        ResultSet rs = pstmt.executeQuery();

        while (rs.next()) {
            //int id_intisari = rs.getInt("id_intisari");
            int id_tweet = rs.getInt("id_tweet");
            String sentimen = rs.getString("sentimen");
            float prob = rs.getFloat("prob");

            SQL = "INSERT INTO mastertweet (sentimen) VALUES (?)";
            ConnectDatabaseClass db1 = new ConnectDatabaseClass();
            Connection con = db1.getConnection();
            pstmt = con.prepareStatement(SQL);
            pstmt.setString(1, sentimen);
            pstmt.executeUpdate();
            db1.closeConnection();
            //selProbs.add(probs);
        }
    }
}

```

```

    } catch (SQLException ex) {
        Logger.getLogger(ConnectDatabaseClass.class.getName()).log(Level.SEVERE, null,
ex);
    }
}

```

```

public int countRows(int id_tweet, String sentimen) throws SQLException {
    // select the number of rows in the table
    String SQL = "SELECT * FROM tweetngram where sentimen = ? and id_tweet = ?";
    PreparedStatement pstmt = null;
    ResultSet rs = null;
    int rowCount = -1;
    try {
        ConnectDatabaseClass db = new ConnectDatabaseClass();
        Connection connections = db.getConnection();
        pstmt = connections.prepareStatement(SQL);
        pstmt.setString(1, sentimen);
        pstmt.setInt(2, id_tweet);
        // get the number of rows from the result set
        rs.next();
        rowCount = rs.getInt(1);
        db.closeConnection();
    } finally {
        rs.close();
        pstmt.close();
    }
    return rowCount;
}
}

```

LAMPIRAN E

Lampiran ini berisi syntax pembuata database pada MySQL.

```
CREATE TABLE intisari (  
    tweetmaster_id_tweet INTEGER UNSIGNED NOT NULL,  
    id_intisari INTEGER UNSIGNED NOT NULL AUTO_INCREMENT,  
    sentimen VARCHAR(10) NULL,  
    prob FLOAT NULL,  
    PRIMARY KEY(tweetmaster_id_tweet, id_intisari),  
    INDEX Table_04_FKIndex1(tweetmaster_id_tweet)  
);
```

```
CREATE TABLE probabilitas (  
    id_prob INTEGER UNSIGNED NOT NULL AUTO_INCREMENT,  
    tweetngram_id_ngram INTEGER UNSIGNED NOT NULL,  
    tweetngram_tweetmaster_id_tweet INTEGER UNSIGNED NOT NULL,  
    ngram VARCHAR(255) NULL,  
    qty INTEGER UNSIGNED NULL,  
    sentimen VARCHAR(10) NULL,  
    prob FLOAT NULL,  
    PRIMARY KEY(id_prob, tweetngram_id_ngram, tweetngram_tweetmaster_id_tweet),  
    INDEX probabilitas_FKIndex1(tweetngram_id_ngram, tweetngram_tweetmaster_id_tweet)  
);
```

```
CREATE TABLE sentimen (  
    kata VARCHAR(255) NULL,  
    sentimen VARCHAR(10) NULL  
);
```

```
CREATE TABLE stoplist (  
  id_stop INTEGER UNSIGNED NOT NULL AUTO_INCREMENT,  
  stopword VARCHAR(255) NULL,  
  PRIMARY KEY(id_stop)  
);
```

```
CREATE TABLE tweetmaster (  
  id_tweet INTEGER UNSIGNED NOT NULL AUTO_INCREMENT,  
  user_2 VARCHAR(255) NULL,  
  tweet VARCHAR(255) NULL,  
  sentimen VARCHAR(10) NULL,  
  PRIMARY KEY(id_tweet)  
);
```

```
CREATE TABLE tweetngram (  
  id_ngram INTEGER UNSIGNED NOT NULL AUTO_INCREMENT,  
  tweetmaster_id_tweet INTEGER UNSIGNED NOT NULL,  
  ngram VARCHAR(255) NULL,  
  PRIMARY KEY(id_ngram, tweetmaster_id_tweet),  
  INDEX tweetngram_FKIndex1(tweetmaster_id_tweet)  
);
```