# BAB II
# KAJIAN LITERATUR

## 2.1 Pendahuluan

Pada bab sebelunya sudah dijelaskan pada rumusan masalah dari penelitian ini adalah untuk mengetahui kinerja algoritma *k- Nearest Neighbor* dan *Modified k-Nearest Neighbor* pada klasifikasi data menggunakan dataset *website phishing.* Berdasarkan rumusan masalah tersebut penulis akan melakukan analisis terhadap kinerja dari algoritma *k- Nearest Neighbor* dan *Modified k- Nearest Neighbor,* analisis dilakukan terhadap nilai *precision, recall, accuracy,* waktu komputasi dan besarnya memori yang dibutuhkan kedua algoritma untuk klasifikasi data. Pada bab ini penulis akan melakukan literature review dari jurnal, buku, artikel yang terkait dengan metode yang digunakan pada penelitian ini yaitu algoritma klasifikasi k- Nearest Neighbor dan Modified k- Nearest Neighbor serta evaluasi kinerja algoritma.

## 2.2 Landasan Teori

### 2.2.1 *Websites Phishing*

Menurut (James, 2006) Kata *phishing* berasal dari frasa *"website phishing"* merupakan variasi dari kata *"fishing".* Idenya adalah umpan dibuang dengan harapan bahwa pengguna akan mengambilnya dan menggigitnya seperti ikan. Dalam kebanyakan kasus, umpan adalah email atau website pesan instan, yang akan membawa pengguna ke *website phishing.*

Berdasarkan artikel yang berjudul *Phishing Websites Features* oleh (Mohammad, Thabtah, & McCluskey, 2015) pada artikel tersebut terdapat 30 atribut dan 2 kelas target yang dinyatakan sebagai *phishing websites features* yang akan digunakan sebagai *dataset* pada penelitian ini, 30 atribut tersebut yaitu; having_IP_Address, URL_Length, Shortining_Service, having_At_Symbol, double_slash_redirecting, Prefix_Suffix, having_Sub_Domain, SSLfinal_State, Domain_registeration_length, Favicon, port, HTTPS_token, Request_URL, URL_of_Anchor, Links_in_tags, SFH, Submitting_to_email, Abnormal_URL, Redirect, on_mouseover, RightClick, popUpWidnow, Iframe, age_of_domain, DNSRecord, web_traffic, Page_Rank, Google_Index, Links_pointing_to_page Statistical_report dan Result.

### 2.2.2 Data Mining

Data mining adalah proses menemukan pola yang menarik dari sejumlah data. Sebagai proses penemuan pengetahuan, biasanya melibatkan *data cleaning, data integration, data selection, data transformation, pattern discovery, pattern evaluation,* dan *knowledge presentation.* Data mining dapat dilakukan pada semua jenis data yang berasal dari *database, data warehouse, web, repository* informasi selama data tersebut memiliki makna dan dapat dialirkan ke dalam sistem secara dinamis (Han, Pei, & Kamber, 2011).

### 2.2.3 Klasifikasi

Klasifikasi adalah proses menemukan sebuah model atau pola yang menggambarkan dan membedakan kelas atau konsep pada data, model diturunkan

berdasarkan analisis dari set data latih, model ini digunakan untuk memprediksi label kelas objek yang label kelasnya tidak diketahui (Han et al., 2011). Menurtut (Zaki, Meira Jr, & Meira, 2014) klasifikasi dapat didefinisikan sebagai proses untuk menyatakan suatu objek data sebagai salah satu kategori (kelas) yag telah didefinisikan sebelumnya .

Klasifikasi sendiri merupakan salah satu tugas dari data mining yang bertujuan untuk memprediksi label kategori benda yang tidak diketahui sebelumnya, dalam membedakan antara objek yang satu dengan yang lainnya berdasarkan atribut atau fitur. (Wafiyah et al., 2017) untuk melakukan klasifikasi data terdapat banyak algoritma yang dapat digunakan untuk mengklasifikasikan data antara lain *k-Nearest Neighbor, support vector machine, fuzzy* dan *artificial neural network*.

### 2.2.4 Algoritma *k- Nearest Neighbor*

Algoritma *k- Nearest Neighbor* atau disingkat kNN adalah suatu algoritma yang menggunakan pendekatan *supervised learning.* Perbedaan antara *supervised learning* dengan *unsupervised learning* yaitu *supervised learning* bertujuan untuk menemukan pola baru pada data dengan menghubungkan pola data yang lama data yang baru. Sedangkan *unsupervised learning*, data tersebut belum memiliki pola apapun, sehingga tujuan *unsupervised learning* adalah menemukan pola dalam data.

Menurut (Han et al., 2011) *k- Nearest Neighbor* merupakan algoritma *Nearest-Neighbor Classifiers* yang didasarkan pada pembelajaran, analoginya

dengan mencari jarak antara dua titik data yaitu titik data *training* dan titik data *testing*, setiap data *testing* dan data *training* mewakili titik dalam ruang n-dimensi, semua data *training* disimpan dalam ruang pola n-dimensi, ketika diberi data yang tidak diketahui yang digunakan sebagai data *testing*, maka *k- Nearest Neighbor* akan mencari ruang pola pada data *training* yang memiliki kedekatan paling dekat dengan nilai *k* tetangga terdekat dari data yang tidak diketahui tersebut.

Menurut (Bhatia, 2010) keunggulan dari algoritma *k- Nearest Neighbor* yaitu waktu pelatihan sangat cepat, sederhana dan mudah dipelajari, tahan terhadap data pelatihan yang memiliki derau, dan efektif jika data pelatihan besar, sedangkan kekurangannya sendiri yaitu nilai k bias, komputasi kompleks, keterbatasan memori, dan mudah tertipu dengan atribut yang tidak relevan.

(Karegowda, Jayaram, & Manjunath, 2012) Langkah kerja dari algoritma *k-Nearest Neighbor* untuk mengklasifikasikan data yaitu :

1. Mendefinisikan nilai *k.*
2. Melakukan perhitungan nilai jarak data training dan testing.
3. Mengelompokan data berdasarkan hasil hitung jarak..
4. Mengelompokkan data berdasarkan tetangga terdekat.
5. Memilih nilai yang sering muncul dari tetangga terdekat sebagai prediksi data selanjutnya.

### 2.2.5 Perhitungan *k* Terbaik

Pada algoritma *k-Nearest Neighbor* hal yang paling penting adalah bagaimana menentukan parameter *k* yang terbaik/optimum, nilai *k* yang terlalu

tinggi akan menjadikan batasan antara class menjadi semakin samar sedangkan nilai *k* yang terlalu kecil membuat kurang sensitive terhadap data yang ada. Berdasarkan pernyataan (Islam, Wu, Ahmadi, & Sid-Ahmed, 2007) bahwa parameter *k* haruslah berupa bilangan ganjil karena harus memilih ketetanggaan terdekat menjadi dua pilihan kelas yaitu (Ya atau Tidak).

Pada penelitiannya (Hassanat, Abbadi, Altarawneh, & Alhasanat, 2014) untuk memecahkan masalah parameter *k* pada algoritma *k-Nearest Neighbor* menggunakan parameter *k* dengan parameter *k* = 1 sampai *k* = akar kuadrat dari data latih (yaitu *k*=1, *k*=3, *k*=5, *k*=7, *k*=9 dan $k = \sqrt{n}$ ) ditambah dengan k=30, *k*=45 dan *k*=60 untuk mewaiki nilai *k* bernilai kecil, menengah dan besar. Pada penelitiannya mengatakan bahwa $k = \sqrt{n}$ digunakan sebagai patokan menentukan nilai *k* bukan sebagai pilihan yang baik untuk pengklasifikasian *k-Nearest Neighbor* karena tidak menghasilkan hasil yang sangat baik dibandingkan lainnya. Serta *k*=30, *k*=45 dan *k*=60 tidak membantu dalam keakuratan algoritma *k-Nearest Neighbor*.

### 2.2.6 Perhitungan Jarak *Euclidean*

*Euclidean distance* merupakan metode menghitung jarak antara dua buah titik dalam *euclidean speace*. *Euclidean distance* digunakan untuk mengukur jarak antara dua objek data yang memiliki atribut numerik serta dapat diterapkan pada data yang berdimensi 1, 2 dan 3 dimensi, serta sederhana jika diterapkan pada dimensi yang lebih tinggi.

Pada penelitian ini perhitungan jarak menggunakan *eucledian distance* yang dilakukan untuk menghitung nilai jarak antar data latih dan data uji. *Euclidean distance* biasa disebut sebagai jarak garis lurus. Pada persamaan 2 .1 dapat dilihat perumusan *euclidean distance* dimana x merupakan titi titik pada data training dan y merupakan titik pada data testing, untuk mencari jarak antara dua titik tersebut dirumuskan :

$$d(x,y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2} \quad (2.1)$$

Keterangan:

*d* : Jarak antara titik pada data training *x* dan titik data testing *y*

*x* : Merepresentasikan data training ke *i* dinyatakan *( $x_1$, $x_2$, $x_3$, …., $x_i$ ).*

*y* : merepresentasikan data testing ke *i* dinyatakan *( $y_1$, $y_2$, $y_3$, …., $y_i$ ).*

### 2.2.7 Algoritma *Modified k- Nearest Neighbor*

Pada dasarnya algoritma *Modified k- Nearest Neighbor* atau disingkat MkNN adalah algoritma hasil pengembangan dari algoritma *k- Nearest Neighbor* yang di modifikasi sebagai solusi dari kelemahan yang ada pada algoritma k-Nearest Neighbor tardisional dengan penambahan beberapa proses yaitu, perhitungan nilai validitas dan *wight voting.* Menurut (Parvin et al., 2010) modifikasi dilakukan pada algoritma *k- Nearest Neighbor* bertujuan untuk meningkatkan kinerja dari algoritma *k- Nearest Neighbor* dan mengatasi masalah terhadap tingkat akurasi yang rendah dari algoritma *k- Nearest Neighbor* tradisional.

Berdasarkan penelitian (Wafiyah et al., 2017) langkah kerja dari algoritma *Modified k- Nearest Neighbor* untuk mengklasifikasikan data yaitu :

1. Menentukan terlebih dahulu data latih dan data uji serta nilai *k.* yang akan digunakan.
2. Melakukan perhitungan jarak antara data latih dengan perhitungan *euclidean distance.*
3. Melakukan perhitungan nilai validitas

   Proses perhitungan nilai validasi dilakukan untuk semua data pada data latih. Nilai validitas setiap data tergantung pada setiap tetangganya atau nilai *k.* Setelah dihitung validitas tiap data maka nilai validitas tersebut digunakan sebagai informasi lebih mengenai data tersebut. Persamaan untuk menghitung nilai validitas pada setiap data training dapat dilihat pada persamaan 2.2.

$$Validity\ (x) = \frac{1}{k}\sum_{i=0}^{n} S(label(x), label\big(Ni(x)\big)) \tag{2.2}$$

   Keterangan:

   *i* : jumlah data latih

   *k* : jumlah tetangga terdekat data latih dari similaritas yang terbaik.

   *Ni(x)* : Banyaknya label kelas titik terdekat x

   *S* : Fungsi menghitung nilai kesamaan antara titi x dan data ke –i dari tetangga terdekat yang dapat dilihat pada persamaan 2.3.

$$S(a,b) = \begin{cases} 1\ a = b \\ 0\ a \neq b \end{cases} \tag{2.3}$$

Dimana :

*a* : kelas a pada data training.

*b* : kelas selain a pada data training.

4. Menghitung jarak antara data latih dengan data uji yang ditetapkan dengan menggunakan Euclidean distance.
5. Menghitung nilai weighted volting

   Pada algoritma Modified k-Nearest Neighbor perhitungan nilai weighted volting berdasarkan nilai validitas dan jarak antara data latih dan data uji, perhitungan weighted volting dapat dilihat pada persamaan 2.4.

$$W(i) = Validity\ (i)\ x\ \frac{1}{de+0,5} \quad (2.4)$$

   Keterangan:

   $(i)$ : Nilai Validitas

   $de$ : *different(x)* atau Jarak data latih

   $W$ : Merepresentasikan bobot antara data uji dengan data latih ke-$i$
6. Mengurutkan nilai weighted voting secara descending
7. Menentukan label dengan nilai bobot terbesar yang kemudian akan ditetapkan sebagai label klasifikasi data uji.

### 2.2.8 *k-Fold Cross Validation*

Menurut (Refaeilzadeh, Tang, & Liu, 2009) c*ross validation* merupakan metode statistik untuk mengevaluasi dan membandingkan algoritma pembelajaran

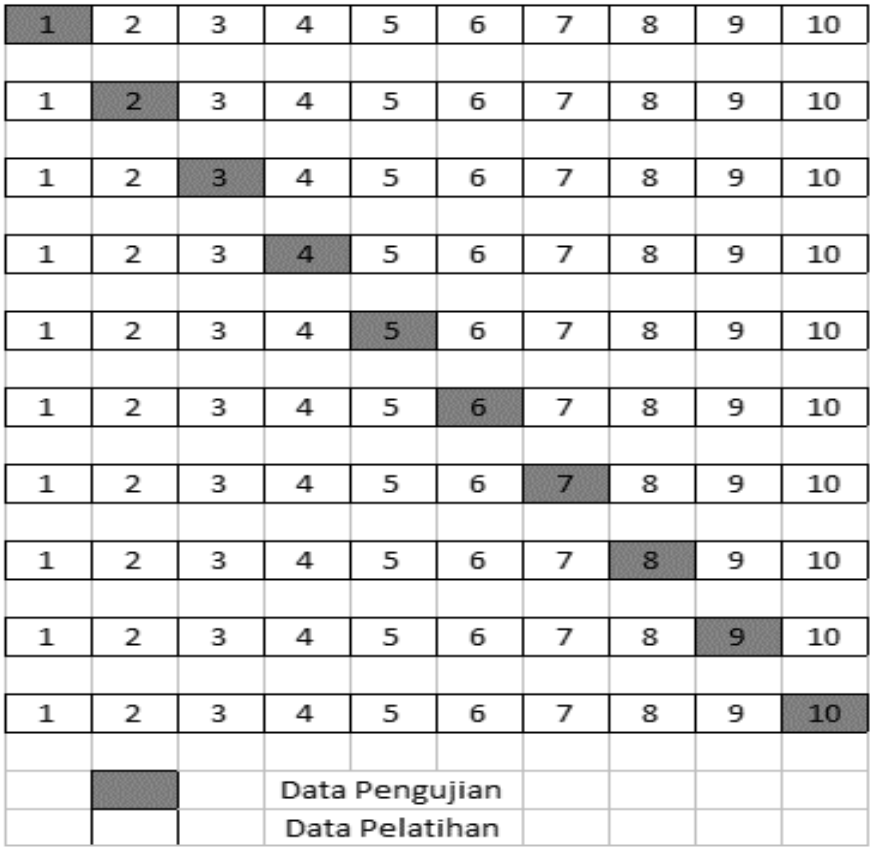
dengan membagi data menjadi dua segmen: satu digunakan untuk mempelajari atau melatih model dan yang lainnya digunakan untuk memvalidasi model.

Pada *cross validation* terdapat dua pendekatan populer untuk mengevaluasi kinerja algoritma yaitu *k-fold cross validation* dan *leave-one-out cross validation.* Ketika jumlah data besar, *k-fold cross validation* harus digunakan untuk memperkirakan akurasi data (Frank & Hall, 2011).

Kerja dari *k-fold cross-validation* (Olson & Delen, 2008) data pertama-tama dipartisi menjadi *k* atau segmen yang berukuran sama (atau hampir sama). Selanjutnya dilakukan sejumlah *k*-kali validasi dengan masing-masing validasi menggunakan data partisi ke-*k* sebagai data testing dan menggunakan sisa partisi lainnya sebagai data training, tahap selanjutnya menghitung rata-rata akurasi dari *k*-kali validasi yang digunakan sebagai validasi final. Dalam data mining dan machine learning 10-*fold cross-validation* adalah yang paling umum digunakan (Refaeilzadeh et al., 2009).

(Han et al., 2011) juga secara umum merekomendasikan 10-*fold cross validation* untuk memperkirakan akurasi (bahkan jika daya komputasi memungkinkan menggunakan lebih *folds*) karena bias dan varians yang relatif rendah.

Pada Gambar II-1 mengilustrasikan sekema dari 10-*fold cross validation.*, data dibagi menjadi 10 *fold* berukuran yang sama, sehingga kita memiliki 10 subset data untuk mengevaluasi kinerja dari algoritma, untuk masing-masing dari 10 subset data tersebut *cross validation* akan menggunakan 9 fold untuk data latih dan 1 *fold* untuk data uji.




Gambar II-1. Skema 10-*fold cross validation.*

### 2.2.9 Pengujian Kinerja Algoritma

Pengujian kinerja algoritma dilakukan untuk mengetahui seberapa baik kinerja dari algoritma klasifikasi dengan menghitung nilai accuracy, recall dan precision dengan menggunakan *confusion matrix*. Menurut (Han et al., 2011) *Confusion matrix* adalah *matrix* yang digunakan untuk menguji kinerja dari sebuah klasifikasi yang berguna untuk menganalisis seberapa baik *classifier* dapat mengenali data dari berbagai kelas. Untuk masalah dua kelas *True Positive* (TP) dan *True Negative* (TN) menunjukkan ketika *classifier* melakukan sesuatu dengan

benar, sementara *False Positive*(FP) dan *False Negative* (FN) menunjukkan ketika *classifier* melakukan kesalahan. Dengan menggunakan *confusion matrix*, parameter yang akan digunakan untuk mengukuran kinerja algoritma, yaitu *precision, recall*, dan *accuracy*. Secara jelas dilihat pada table II-1 dan persamaan 2.5, 2.6 dan 2.7.

Tabel II-1. C*onfusion matrix.*

| Kelas | Terklasifikasi positif | Terklasifikasi negatif |
|---|---|---|
| Positif | TP | FN |
| Negative | FP | TN |

$$Precision = \frac{TP}{TP+FP} \quad (2.5)$$

$$Recall = \frac{TP}{TP+FN} \quad (2.6)$$

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \; x \; 100\% \quad (2.7)$$

Keterangan:

- TP adalah *True Positive*, yaitu jumlah data positif yang terklasifikasi dengan benar oleh sistem.
- TN adalah *True Negative*, yaitu jumlah data negatif yang terklasifikasi dengan benar oleh sistem.
- FN adalah *False Negative*, yaitu jumlah data negatif namun terklasifikasi salah oleh sistem.
- FP adalah *False Positive*, yaitu jumlah data positif namun terklasifikasi salah oleh sistem.

Selain mengetahui nilai *precision, recall,* dan *accuracy*, pada penelitian ini juga akan mengetahui besarnya waktu komputasi yang dibutuhkan sistem dalam melakukan proses klasifikasi data dan bersarnya memori yang dibutuhkan sistem dalam melakukan klasifikasi data, perhitungan tersebut bertujuan untuk mengetahui kinerja kedua algoritma pada klasifikasi dataset *website phishing.*

## 2.3 Penelitian Lain yang Relevan

Pada bagian ini dipaparkan beberapa penelitian yang telah dilakukan oleh beberapa peneliti lain. Dimaksudkan untuk memperkuat penalaran dan rasionalitas keterlibatan sejumlah variabel pada penelitian ini. Selain itu juga difungsikan sebagai pendapat ilmilah yang dipadukan dengan hasil kajian pustaka untuk membangun kerangka berpikir peneliti dalam kaitannya dengan masalah yang sedang diteliti.

### 2.3.1 A Modification on k- Nearst Neighbor

Penelitian (Parvin et al., 2010) untuk meningkatkan kinerja dari algoritma kNN mereka menambahkan proses validitas dan Perhitungan Weighted Voting untuk mengkalsifikasikan 9 *dataset* yaitu Monk1, Monk2, Monk3, Isodata, Wine, Iris, Balanc-sc, Bupa, dan SAHeart. Dengan ditambah validitas dan Perhitungan Weighted Voting pada algoritma *K-Nearst Neighbor* dengan menggukanan nilai parameter k yang berbeda-beda 3, 5 dan 7. Sehingga dari *dataset* yang digunakan didaptkan hasil algoritma mengalami peningkatan dari tingkat akurasinya untuk setiap *dataset* yang ada. Sehingga berdasarkan 9 *dataset* tersebut algoritma MkNN mengungguli algoritma tradisional kNN terutama dari segi ketahanan dan akurasi.

2.3.2 Implementasi algoritma modified k- Nearst Neighbor untuk Klasifikasi Penyaikit demam.

Penelitian (Wafiyah et al., 2017) menerapkan algoritma *Modified k-Nearest Neighbor* untuk mengkalsifikasi penyakit penyakit demam, penelitian tersebut dilakukan beberapa pengujian yaitu pengujian terhadap perubahan nilai *k*, pengujian terhadap perubahan jumlah data latih dan pengujian terhadap perubahan komposisi antara data latih, sehingga didapatkan hasil rata-rata akurasi untuk pengujian pengaruh nilai *k* terhadap akurasi sebesar 88.55%, hasil rata-rata akurasi untuk pengujian pengaruh variasi jumlah data latih adalah 92.42%, hasil rata-rata akurasi untuk pengujian pengaruh komposisi data latih adalah 87.89% dan hasil pengujian pengaruh komposisi data latih dan data uji terhadap akurasi mendapatkan nilai rata-rata akurasi sebesar 96.35%.

2.3.3 Analisis Komparatif Evaluasi Performa Algoritma Klasifikasi pada Readmisi Pasien Diabetes.

Penelitian (Yusa et al., 2016) Analisis dilakukan menggunakan algoritma *Decision Tree, k- Nearst Neighbor* (k-NN), dan *Naïve Bayes*, penelitian mnggunakan dataset dari UCI *Machine Learning* dan data dilakukn Preparasi data dari 55 atribut dan 101.776 *records* data menjadi 47 atribut dan 49.735 *records* data. Selanjutnya data akan diproses menggunakan 3 algoritma yang digunakan untuk menghitung performa *accuracy, Mean Absolute Error (MAE*) dan *Kappa Satatic* dengan model 10-*Fold Crss Validation*. Hasil akhir diketahui bahwa

algoritma *Naïve Bayes* memiliki tingkat akurasi 57.52%, MAE sebesar 0.512, dan kappa statistic sebesar 0.182. sehingga dinyatakaan algoritma naïve bayes memiliki performa lebih baik dari *Decision Tree, k-Nearst Neighbor*.

### 2.3.4 Comparative Analysis of k- Nearst Nighbor and Modified k- Nearst Neighbor Algorithm for Data Classification

Penelitian (Gazalba & Reza, 2017) perbandingan yang dilakukan pada penelitian ini yaitu membandingkan nilai akurasinya yang dihitung menggunakan *confusion matrix*, dataset yang digunakan yaitu data kalsifikasi bantuan tunai bersarat (unit pelaksanaan program keluarga harapan) sebanyak 7395 *records* data. Nilai *k* pada k-*fold cross validation* untuk penelitian ini digunakan 3-*fold cross validation* dan parameter *k* sebanyak 10, hasil penelitian ini didapatkan bahwa algoritma *Modified k-Nearst Neighbor* memiliki keunggul nilai akurasi yang lebih baik dari pada *k-Nearst Neighbor* biasa berdasarkan *dataset* yang digunakan.

## 2.4 Kesimpulan

Berdasarkan uraian diatas telah dijelaskan mengenai hal-hal yang berkaitan dengan penelitian yang akan dilakukan, penjelasan mengenai tahapan pada algoritma *k-Nearst Nighbor*, algoritma *Modified k-Nearst Neighbor* dan perhitungan evaluasi terhadap kinerja algoritma serta dijelaskan beberapa penelitian terdahulu yang relevan dengan penelitian yang akan penulis teliti pada penelitian ini.