APLIKASI PENDETEKSI *WEBSITE PHISHING* MENGGUNAKAN *MACHINE LEARNING*

TUGAS AKHIR

Disusun dalam rangka memenuhi salah satu persyaratan

Untuk menyelesaikan program Strata-1 Departemen Teknik Informatika

Fakultas Teknik Universitas Hasanuddin

Makassar

Disusun Oleh:

DIKI WAHYUDI

D421 15 518

DEPARTEMEN TEKNIK INFORMATIKA

FAKULTAS TEKNIK

UNIVERSITAS HASANUDDIN

MAKASSAR

2020

LEMBAR PENGESAHAN

"APLIKASI PENDETEKSI WEBSITE PHISHING MENGGUNAKAN MACHINE LEARNING"

Disusun Oleh:

DIKI WAHYUDI **D421 15 518**

Skripsi ini telah dipertahankan pada Ujian Akhir Sarjana tanggal 04 Januari 2020. Diterima dan disahkan sebagai salah satu syarat memperoleh gelar Sarjana Teknik (S.T) pada Departemen Teknik Informatika Fakultas Teknik Universitas Hasanuddin.

Gowa, 07 Januari 2020

Disetujui Oleh:

Pembimbing I,

Dr. Eng. Muhammad Niswar, ST., M.IT.

NIP. 19730922 199903 1 001

Pembimbing II,

A. Ais Prayogi Alimuddin, ST., M.Eng.

NIP. 19830510 201404 1 001

Diterima dan disahkan oleh:

Ketua Departemen S1 Teknik Informatika

Dr. Amil Ahmad Ilham, S.T., M.IT

NIP. 19731010 199802 1 001

KATA PENGANTAR

Puji dan syukur penulis panjatkan kepada Tuhan yang Maha Esa karena berkat rahmat dan karunia-Nya sehingga tugas akhir yang berjudul "APLIKASI PENDETEKSI WEBSITE PHISHING MENGGUNAKAN MACHINE LEARNING" ini dapat diselesaikan sebagai salah satu syarat dalam menyelesaikan jenjang Strata-1 pada Departemen Teknik Informatika Fakultas Teknik Universitas Hasanuddin.

Penulis menyadari bahwa dalam penyusunan dan penulisan laporan tugas akhir ini tidak lepas dari bantuan, bimbingan serta dukungan dari berbagai pihak, dari masa perkuliahan sampai dengan masa penyusunan tugas akhir. Oleh karena itu, penulis dengan senang hati menyampaikan terima kasih kepada:

1. Kedua Orang tua penulis, Bapak Alimuddin dan Ibu Suriani yang selalu memberikan dukungan, doa, dan semangat serta selalu sabar dalam mendidik penulis sejak kecil;
2. Bapak Dr. Eng. Muhammad Niswar, ST., M.IT selaku pembimbing I dan Bapak A. Ais Prayogi Alimuddin, ST., M.Eng. selaku pembimbing II yang selalu menyediakan waktu, tenaga, pikiran dan perhatian yang luar biasa untuk mengarahkan penulis dalam penyusunan tugas akhir;
3. Bapak Dr. Eng. Ady Wahyudi Paundu, S.T., M.T. yang senantiasa memberikan nasehat, masukan, serta perhatian yang luar biasa kepada penulis dalam penyusunan tugas akhir;

4. Bapak Dr. Amil Ahmad Ilham, ST., M.IT., selaku Ketua Departemen Teknik Informatika Fakultas Teknik Universitas Hasanuddin atas bimbingannya selama masa perkuliahan penulis;
5. Para sahabat dan teman-teman yang telah memberikan begitu banyak bantuan selama penelitian, pengambilan data dan diskusi progress penyusunan tugas akhir;
6. Teman-teman Hypervisor FT-UH atas dukungan dan semangat yang diberikan selama ini;
7. Segenap Staf dan Dosen Departemen Teknik Informatika Fakultas Teknik Universitas Hasanuddin yang telah membantu penulis.
8. Orang-orang berpengaruh lainnya yang tanpa sadar telah menjadi inspirasi penulis.

Akhir kata, penulis berharap semoga Tuhan berkenan membalas segala kebaikan dari semua pihak yang telah banyak membantu. Semoga Tugas Akhir ini dapat memberikan manfaat bagi pengembangan ilmu.

Makassar, 06 November 2019

Penulis

ABSTRAK

Phishing merupakan tindakan untuk mendapatkan informasi penting seseorang berupa username, password, dan informasi sensitif lainnya dengan memberikan *website* palsu yang mirip dengan aslinya. *Phishing* (memancing informasi penting) adalah suatu bentuk tindakan kriminal yang bermaksud untuk mendapatkan informasi rahasia dari seseorang, seperti username, password dan kartu kredit, dengan menyamar sebagai orang atau bisnis yang tepercaya dalam sebuah komunikasi elektronik resmi, seperti surat elektronik atau pesan instan. Seiring dengan perkembangan penggunaan media elektronik, yang diikuti dengan meningkatnya pula *cyber crime* seperti salah satunya serangan *phishing* ini. Oleh karena itu, untuk meminimalisir serangan *phishing* dibutuhkan sebuah sistem yang dapat mendeteksi serangan tersebut. *Machine Learning* merupakan salah satu metode yang dapat digunakan untuk membuat sistem yang dapat mendeteksi *phishing.* Data yang digunakan dalam penelitian ini sebanyak 11055 data website, yang terbagi atas dua *class* yaitu "*legitimate*" dan "*phishing*". Data ini kemudian dibagi dengan menggunakan *10-fold cross validation*. Sedangkan algoritma yang digunakan adalah algoritma *Support Vector Machine* (SVM) yang dibandingkan dengan algoritma *decision tree* dan *k-nearest neighbor* dengan melakukan optimasi parameter pada setiap algoritma. Dari hasil pengujian pada penelitian ini diperoleh akurasi sistem terbaik 85.71% menggunakan SVM *kernel polynomial* dengan nilai *degree* 9 dan *C 2.5*.

Kata Kunci: *phishing*, *machine learning*, *support vector machine* (SVM), ekstraksi fitur, optimasi parameter.

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I

PENDAHULUAN

1.1. Latar Belakang

Seiring perkembangan teknologi khususnya teknologi informasi seperti Internet yang telah membawa manusia menuju zaman yang berbeda, yaitu zaman yang terasa lebih mudah, praktis, simpel, dan dinamis. Dengan berkembangnya teknologi informasi orang dapat dengan mudah serta cepat dalam mendapatkan informasi apa yang diinginkan bahkan dalam mendapatkan informasi tersebut ada beberapa pihak yang melakukannya dengan cara ilegal serta merugikan orang lain. Internet menjadi bagian penting dalam kehidupan masyarakat. Internet telah mempengaruhi seluruh bagian kehiduapan masyarakat khususnya kehidupan social dan finansial. Sebagai contoh media sosial dan *website* yang digunakan untuk komunikasi dan bisnis. Namun seiring dengan perkembangan teknologi informasi tersebut telah membawa kekhawatiran tersendiri terhadap masyarakat, dimana adanya beberapa pihak yang tidak bertanggung jawab yang dapat merugikan orang lain.

Phishing merupakan tindakan untuk mendapatkan informasi penting seseorang berupa *username, password,* dan informasi sensitif lainnya dengan memberikan *website* palsu yang mirip dengan aslinya. *Phishing* (memancing informasi penting) adalah suatu bentuk tindakan kriminal yang bermaksud untuk tkan informasi rahasia dari seseorang, seperti *username, password* dan dit, dengan menyamar sebagai orang atau bisnis yang tepercaya dalam omunikasi elektronik resmi, seperti surat elektronik atau pesan instan.

Istilah *phishing* dalam bahasa Inggris berasal dari kata fishing ('memancing'), dalam hal ini berarti memancing informasi keuangan dan kata sandi pengguna. Dengan banyaknya kasus penipuan yang dilaporkan, metode tambahan atau perlindungan sangat dibutuhkan. Upaya-upaya itu termasuk pembuatan undang-undang, pelatihan pengguna, dan langkah-langkah teknis. *Phishing* biasanya susah dideteksi khususnya bagi masyarakat awam yang tidak bergerak di bagian teknikal. Hal ini tambah diperparah dengan meningkatnya penggunaan smartphone yang biasanya tidak memperlihatkan URL dari sebuah *website* secara keseluruhan. (Halim Z. 2017:72)

Berbicara mengenai *phishing* maka akan dikaitkan juga dengan sosial media yang penggunaannya telah meningkat setiap tahunnya. Dan apabila diperhatikan sosial media merupakan media yang paling penting bagi seorang hacker dalam melancarkan serangannya. Hal ini dikarenakan penggunaan sosial media yang begitu besar serta mudahnya mendapatkan informasi seseorang melalui akun sosial medianya. Serangan *phishing* seperti asal katanya "fishing" yaitu memancing dengan memberikan umpan kepada korban dan menunggu korban untuk mengambil umpan tersebut.

Dari beberapa penelitian yang telah dilakukan maka pada penelitian ini, penulis mencoba untuk membangaun sebuah aplikasi pendeteksi *phishing* menggunakan metode machine learning dengan algoritma klasifikasi *Support achine* (SVM), *Decision Tree*, dan *K-Nearest Neighbors* dengan Bahasa man python.

1.2. Rumusan Masalah

Berdasarkan latar belakang yang telah dipaparkan, maka rumusan masalah pada penelitian ini antara lain:

1. Bagaimana mengekstrak fitur dari sebuah halaman web berbasis URL untuk digunakan dalam proses deteksi *phishing*?
2. Bagaimana membangun sebuah sistem pendeteksi *phishing* menggunakan algoritma SVM?

1.3. Tujuan Penelitian

Tujuan dalam penelitian ini adalah:

1. Mengekstrak fitur dari sebuah halaman website dengan berbasis URL untuk digunakan dalam proses deteksi *phishing*.
2. Membangun aplikasi pendeteksi *phishing* menggunakan algoritma SVM dengan bahasa pemrograman python.

1.4. Manfaat Penelitian

Manfaat yang diharapkan dalam penelitian ini adalah:

1. Bagi masyarakat, dapat menggunakan aplikasi pendeteksi *phishing* ini untuk mengecek apakah *website* yang akan diakses adalah *phishing* atau bukan.
2. Bagi peneliti, dapat digunakan untuk menambah pengetahuan dan sebagai referensi mengenai deteksi *phishing* dengan menggunakan machine learning.

3. Bagi institusi pendidikan, dapat digunakan sebagai referensi dalam pengembangan penelitian topik terkait untuk mempelajari deteksi *phishing* dengan menggunakan machine learning.

1.5. Batasan Masalah

Ruang lingkup pembahasan tugas akhir ini dibatasi hanya mencakup hal-hal berikut:

1. Aplikasi dibangun dalam bentuk *Command Line Interface* (CLI).
2. Bahasa yang digunakan dalam membangun aplikasi pendeteksi *phishing* yaitu bahasa pemrograman python.
3. Output dari aplikasi pendeteksi *phishing* berupa persentase *phishing* dan hasil prediksi *phishing*.

1.6. Sistematika Penulisan

Untuk memberikan gambaran singkat mengenai isi tulisan ini, maka akan diuraikan beberapa tahapan dari penulisan secara sistematis, yaitu:

BAB I PENDAHULUAN

Bab ini menguraikan secara umum mengenai hal yang menyangkut latar belakang, perumusan masalah, batasan masalah, tujuan, dan manfaat penelitian.

BAB II TINJAUAN PUSTAKA

Bab ini berisi teori-teori terkait hal-hal yang mendasari dan berhubungan dengan penelitian, termasuk di dalamnya iridologi, visi komputer, dan metode-metode yang digunakan dalam penelitian.

BAB III METODOLOGI PENELITIAN

Bab ini berisi tentang perencanaan dan proses penerapan algoritma dan metode-metode dalam pengolahan data, mulai dari preprocessing hingga menghasilkan prediksi.

BAB IV HASIL DAN PEMBAHASAN

Bab ini berisi tentang hasil penelitian dan pembahasan terkait pengolahan data yang telah dilakukan yang disertai dengan tabel hasil penelitian.

BAB V PENUTUP

Bab ini berisi tentang kesimpulan yang didapatkan berdasarkan hasil penelitian yang telah dilakukan serta saran untuk pengembangan sistem yang lebih lanjut.

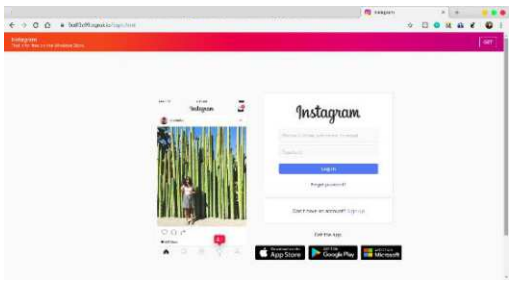
BAB II

TINJAUAN PUSTAKA

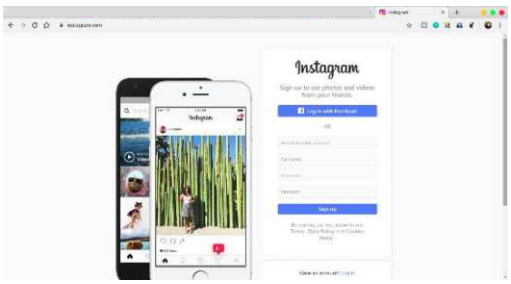
2.1. Phishing

Phishing merupakan tindakan untuk mendapatkan informasi penting seseorang berupa username, password, dan informasi sensitif lainnya dengan memberikan *website* palsu yang mirip dengan aslinya.

Phishing (memancing informasi penting) adalah suatu bentuk tindakan kriminal yang bermaksud untuk mendapatkan informasi rahasia dari seseorang, seperti username, password dan kartu kredit, dengan menyamar sebagai orang atau bisnis yang tepercaya dalam sebuah komunikasi elektronik resmi, seperti surat elektronik atau pesan instan. Dengan banyaknya kasus penipuan yang dilaporkan, metode tambahan atau perlindungan sangat dibutuhkan. Upaya-upaya perlindungan itu termasuk pembuatan undang-undang, pelatihan pengguna, dan langkah-langkah teknis. *Phishing* biasanya susah dideteksi khususnya bagi masyarakat awam yang tidak bergerak di bagian teknikal.[1]



Website Phishing



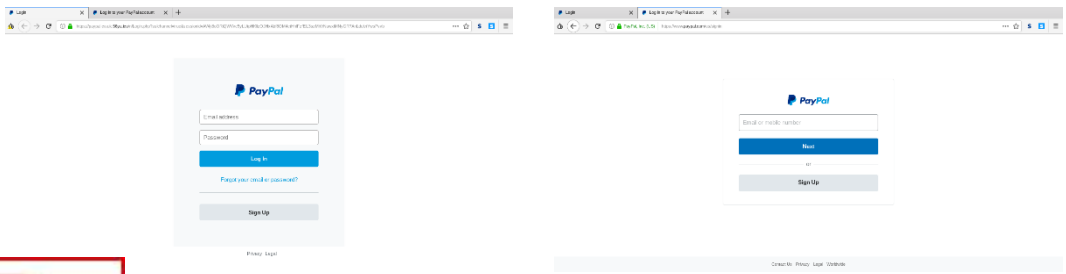
Website Asli

r 2.1. Perbandingan *website phishing* Instagram dengan website aslinya

"Prediksi Website Pemancing Informasi Penting Phising Menggunakan Support Vector VM)", Information System for Educators and Professionals, 2017, hlm. 72.

Serangan *phishing* biasanya berupa sebuah email yang seolah-olah berasal perusahaan resmi, misalnya dari *website-website* yang biasa digunakan oleh *user*. Tujuan serangan ini adalah untuk mendapatkan data-data pribadi *user*, misalnya *password*, nomor rekening, nomor kartu kredit, dan data-data sensitif lainnya.

Seorang penyerang *phishing* biasanya disebut *phisher*. Dalam melakukan serangan, seoarang *phisher* akan menyamar menjadi orang lain yang berasal dari pihak resmi dari sebuah perusahaan dengan cara mengirimkan alamat web yang seolah-olah mengarah pada web resmi dari perusahaan tersebut, padahal alamat web tersebut akan diarahkan ke alamat web yang telah dibuat oleh *phisher* yang menyerupai dengan web resmi perusahaan tersebut. Dengan diarahkannya target ke alamat web yang palsu tersebut maka target akan diminta untuk memasukkan data-data sensitive yang diperlukan oleh *phisher*, seperti diminta untuk memasukkan *username*, *password*, nomor rekening, nomor kartu kredit, dan lain-lain. Seorang *phisher* dalam melakukan serangan *phishing* ini tentunya tidak hanya melakukan serangan terhadap satu target melainkan banyak target, dengan alasan bahwa dari banyak target tersebut akan terdapat satu atau dua target berhasil terpancing dengan serangan *phishing* tersebut.



Website Phishing Website Asli

ar 2.2. Perbandingan *website phishing* PayPal dengan website aslinya

Phishing biasanya memanfaatkan email, *website* palsu, spyware dan berbagai media lainnya untuk melakukan aksinya. Beberapa hal yang menyebabkan aksi *phishing* ini terus terjadi dan memakan banyak korban adalah:[2]

1. Ketidaktahuan atau kurangnya pengetahuan

   Kurangnya pengetahuan akan teknologi komputer membuat pelaku *phishing* mudah mendapatkan mangsanya. Dengan memberikan email yang menakutkan, seperti ancaman hilangnya nama domain akan membuat korbannya segera melakukan apa yang diminta.

2. Tampilan palsu yang menyesatkan

   Pemalsuan *website* dan gambar-gambar sangat mudah dilakukan melalui internet dan pengguna awam biasanya tidak menyadari hal tersebut. Hanya dengan melakukan *copy* dan *paste*, sebuah *website* yang mirip dengan asli akan langsung tercipta. *Phisher* juga bisa membuat *website* yang tampak sangat bagus dengan berbagai komentar pengguna yang semuanya fiktif untuk meyakinkan calon korbannya.

3. Kurangnya perhatian pada indikator keamanan

   Sangat sering, pesan-pesan yang muncul tidak dibaca oleh *user*. Biasanya, pesan-pesan ini terlalu teknis untuk *user* sehingga mereka selalu mengklik tombol "OK" untuk melanjutkan. Kebiasaan semacam ini membawa keuntungan tersendiri untuk *phisher* sehingga mereka

ainal Arifin Al, "Cyber Crime Dalam Bentuk Phising Dalam Undang-Undang Nomor 008 Tentang Informasi Dan Transaksi Elektronik Perspektif Hukum Pidana Islam", Islam Negeri Sunan Ampel, 2016, hlm. 61.

bisa memalsukan *website* dan mendapatkan informasi berharga yang dimasukkan oleh korbannya.

4. Meningkatnya penggunaan platform mobile

   Dengan meningkatnya penggunaan platform mobile ikut meningkatkan pula tindak serangan *phishing*. Hal ini dikarenakan dengan platform mobile seorang *user* kurang memperhatikan alamat web yang kunjungi karena layarnya yang cukup kecil sehingga URL dari web tersebut tidak ditampilkan secara keseluruhan.

Salah bentuk teknik serangan *phishing* adalah *URL Obfuscation.* URL (*Uniform Resource Locator* atau alamat web yang diketik dalam browser untuk membuka suatu *website*) *Obfuscation* adalah suatu teknik menyamarkan alamat URL sehingga tampak tidak mencurigakan untuk pengguna. Adapun macam-macamnya adalah sebagai berikut:[3]

1. String yang menyesatkan

   Memanfaatkan string yang tampak asli seperti adanya kata-kata "Microsoft" atau kata-kata yang umum dikenal. Untuk memalsukan *website* "Microsoft" misalnya, seorang *phisher* akan membuat direktori menggunakan kata-kata yang sama yaitu "Microsoft", seperti http://XX.com/Microsoft.com/freelogin.php pelaku kemudian akan

ainal Arifin Al, "Cyber Crime Dalam Bentuk Phising Dalam Undang-Undang Nomor 008 Tentang Informasi Dan Transaksi Elektronik Perspektif Hukum Pidana Islam", Islam Negeri Sunan Ampel, 2016, hlm. 66.

membuat halaman jebakan untuk mendapatkan *username* dan *password* atau informasi berharga lainnya.

2. Menggunakan simbol "@"

   Simbol "@" sebenarnya digunakan untuk *website* yang membutuhkan autentikasi di mana tanda sebelum simbol "@" menunjukkan *username*, sedangkan setelahnya menunjukkan domain. Contoh sederhana pada email sto@jasakom.com.

   Kata "sto" menunjukkan nama sedangkan "jasakom.com" menunjukkan domain. Teknik ini pernah memakan banyak korban dan pernah sangat populer, yaitu http://www.microsoft.com@www.hacker.com. Domain ini jika di klik oleh pengguna maka akan masuk ke situs www.hacker.com, bukan situs Microsoft yang sebenarnya.

3. Nama yang mirip

   Teknik yang pernah menimpa situs klikbca.com ini akan membuat nama yang mirip dan memanfaatkan kelemahan *user* yang suka salah ketik atau salah ingat. Sebagai contoh pada kasus klikbca.com, *phisher* bisa membuat *website* kilikbca.com, klickbca.com dan lain sebagainya. Tentu saja, alamat palsu ini juga dibuat dengan tampilan yang sama persis dengan situs aslinya. Memanfaatkan nama yang mirip tidak harus selalu memanfaatkan kesalahan ketikan atau kesalahan ingat. *Phisher* juga bisa membuat nama domain yang tampak asli seperti microsoft-online.com, microsoft-user.com, dan lain sebagainya.

4. *Shortener URL*

Layanan *shortener* URL menjadi terkenal dan sering digunakan, sebagai contoh perhatikan URL dari Amazon berikut ini:

"http://www.amazon.com/Kindle-Wireless-Reading-Display-Globally/dp/B003FSUDM4/ref=amb_link_353459562_2?pf_rd_m=ATVPDKIKX0DER&pf_rd_s=center-10&pf_rd_r=11EYKTN682A79=1i=B002Y27P3M"

Menghafal alamat yang sedemikian panjang tentu akan sulit dan bahkan hampir tidak mungkin untuk dilakukan kecuali untuk orang-orang jenius. Kini, dengan bantuan layanan *shortener* URL, alamat contoh di atas bisa berubah menjadi:

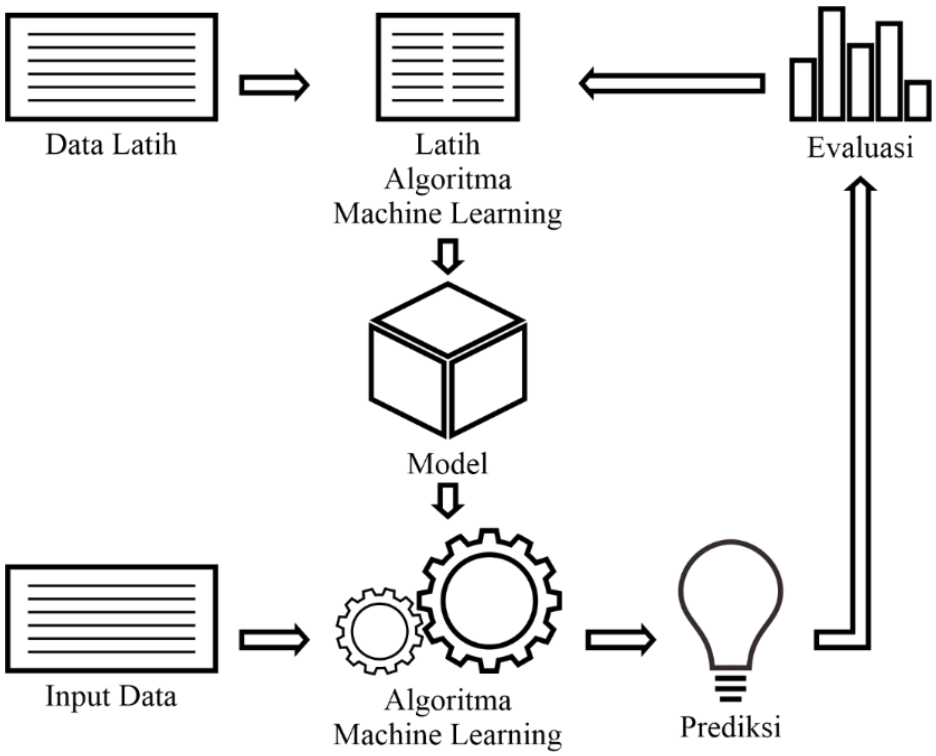
"http://tinyurl.com/KindleWireless"

Karena terbiasa untuk menggunakan layanan semacam ini, banyak orang yang tidak lagi memperhatikan alamat asli yang digunakan. Pelaku phishing bisa memanfaatkan ini untuk menutupi URL asli yang digunakan.

2.2. Machine Learning

Machine learning adalah disiplin ilmu dari *Artificial Intelligence* (Kecerdasan Buatan) yang menggunakan teknik statistika untuk menghasilkan suatu model otomatis dari sekumpulan data yang biasa disebut *dataset*, dengan tujuan memberikan komputer kemampuan untuk "belajar". Pembelajaran mesin hine learning memungkinkan komputer mempelajari sejumlah data (*learn a*) sehingga dapat menghasilkan suatu model untuk melakukan proses

input-output tanpa menggunakan kode program yang dibuat secara eksplisit. Proses belajar tersebut menggunakan algoritma khusus yang disebut *machine learning algorithms*. Terdapat banyak *algoritma machine learning* dengan efesiensi dan spesifikasi kasus yang berbeda-beda. Tidak hanya individu yang belajar meningkatkan kecerdasan nya tetapi mesin juga membutuhkan hal tersebut untuk meningkatkan kecerdasannya dan memiliki kemampuan yang cerdas dan tidak dimiliki oleh mesin lainnya.[4]



Gambar 2.3. Konsep *Machine Learning*

Secara fundamental cara kerja *machine learning* adalah belajar seperti manusia dengan menggunakan contoh-contoh dan setelah itu barulah dapat

[4] ia, "Apa itu Machine Learning dan Cara Kerjanya", diakses dari w.advernesia.com/blog/data-science/machine-learning-adalah/, pada tanggal 10 019 pukul 10.27.

menjawab suatu pertanyaan terkait. Proses belajar ini menggunakan data yang disebut *train dataset* (data latih). Berbeda dengan program statis, *machine learning* diciptakan untuk membentuk program yang dapat belajar sendiri. Gambaran konsep *machine learning* dapat dilihat pada Gambar 2.3.

Dari data tersebut, komputer akan melakukan proses belajar (training) untuk menghasilkan suatu model. Proses belajar ini menggunakan *algoritma machine learning* sebagai penerapan teknik statistika. Model inilah yang menghasilkan informasi, kemudian dapat dijadikan pengetahuan untuk memecahkan suatu permasalahan sebagai proses input-output. Model yang dihasilkan dapat melakukan klasifikasi ataupun prediksi kedepannya.

Untuk memastikan efesiensi model yang terbentuk, data akan dibagi menjadi data latih (*train dataset*) dan data uji (*test dataset*). Pembagian data yang digunakan bervariasi bergantung algoritma yang digunakan. Pada umumnya *train dataset* lebih banyak dari *test dataset*, misalnya dengan rasio 3:1. *Test dataset* digunakan untuk menghitung seberapa efesien model yang dihasilkan untuk melakukan klasifikasi atau prediksi kedepannya yang disebut *test score*. Semakin banyak data yang digunakan, *test score* yang dihasilkan semakin baik. Nilai *test score* bisa berada dalam rentang 0 sampai 1 atau -1 sampai 1.

2.3. *Support Vector Machine* (SVM)

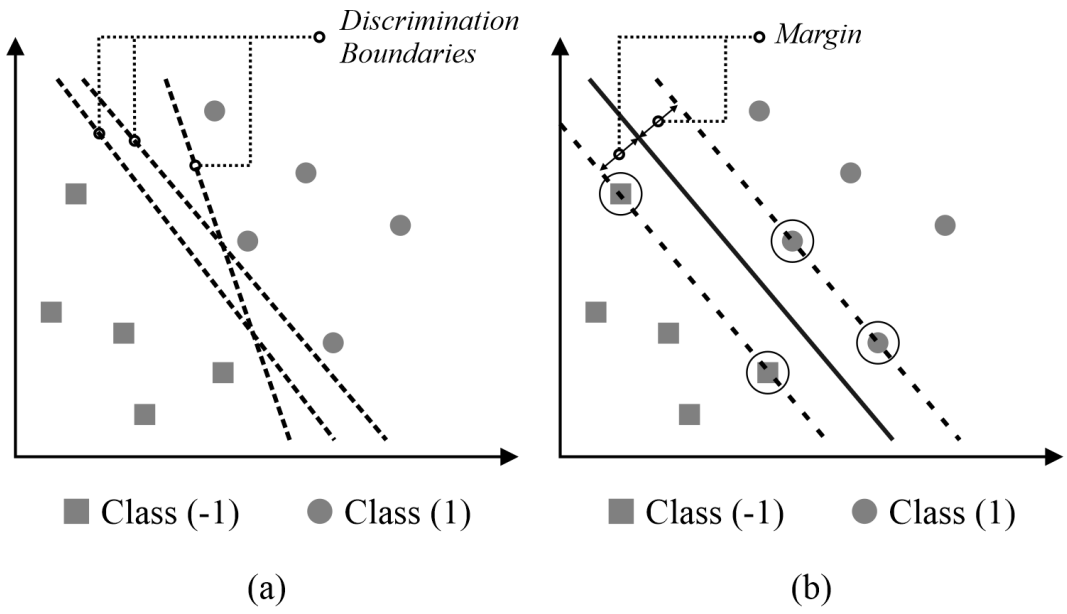
Support Vector Machine (SVM) adalah salah satu metode yang akhir- akhir ini mendapat perhatian. *Support Vector Machine* (SVM) dikembangkan oleh Boser, Guyon, Vapnik, dan pertama kali dipresentasikan pada tahun 1992 di

Annual Workshop on Computational Learning Theory. Konsep dasar SVM sebenarnya merupakan kombinasi harmonis dari teoriteori komputasi yang telah ada puluhan tahun sebelumnya, seperti margin hyperplane (Duda & Hart tahun 1973, Cover tahun 1965, Vapnik 1964, dsb.), kernel diperkenalkan oleh Aronszajn tahun 1950, dan demikian juga dengan konsep-konsep pendukung yang lain. Akan tetapi hingga tahun 1992, belum pernah ada upaya merangkaikan komponen-komponen tersebut. Prinsip dasar SVM adalah *linear classifier*, dan selanjutnya dikembangkan agar dapat bekerja pada problem *non-linear*. dengan memasukkan konsep *kernel trick* pada ruang kerja berdimensi tinggi.[5]

Support Vector Machine (SVM) juga dikenal sebagai teknik pembelajaran mesin (*machine learning*) paling mutakhir setelah pembelajaran mesin sebelumnya yang dikenal sebagai *Neural Network* (NN). Baik SVM maupun NN tersebut telah berhasil digunakan dalam pengenalan pola. Pembelajaran dilakukan dengan menggunakan pasangan data input dan data output berupa sasaran yang diinginkan. Pembelajaran dengan cara ini disebut dengan pembelajaran terarah (*supervised learning*). Dengan pembelajaran terarah ini akan diperoleh fungsi yang menggambarkan bentuk ketergantungan input dan outputnya. Selanjutnya, diharapkan fungsi yang diperoleh mempunyai kemampuan generalisasi yang baik, dalam arti bahwa fungsi tersebut dapat digunakan untuk data input di luar data pembelajaran. diperoleh mempunyai kemampuan generalisasi yang baik, dalam arti bahwa fungsi tersebut dapat digunakan untuk data input di luar data pembelajaran.

[5] Erton F., "Support Vector Machine", Universitas KH. A. Wahab Hasbullah, 2018, hlm.

Konsep SVM dapat dijelaskan secara sederhana sebagai usaha mencari *hyperplane* terbaik yang berfungsi sebagai pemisah dua buah *class* pada *input space*. Gambar 2.4. bagian (a) memperlihatkan beberapa *pattern* yang merupakan anggota dari dua buah *class*: positif (dinotasikan dengan 1) dan negatif (dinotasikan dengan -1). *Pattern* yang tergabung pada *class* negatif disimbolkan dengan kotak, sedangkan *pattern* pada *class* positif, disimbolkan dengan lingkaran. Proses pembelajaran dalam problem klasifikasi diterjemahkan sebagai upaya menemukan garis (*hyperplane*) yang memisahkan antara kedua kelompok tersebut. Berbagai alternatif garis pemisah (*discrimination boundaries*) ditunjukkan pada gambar 2.4. bagian (a).[6]



Gambar 2.4. SVM Berusaha Menemukan *Hyperplane* Terbaik

Hyperplane pemisah terbaik antara kedua *class* dapat ditemukan dengan r *margin hyperplane* dan mencari titik maksimalnya. *Margin* adalah jarak

[6] Erton F., "Support Vector Machine", Universitas KH. A. Wahab Hasbullah, 2018, hlm.

antara *hyperplane* tersebut dengan data terdekat dari masing-masing *class*. Subset data training set yang paling dekat ini disebut sebagai support vector. Garis solid pada Gambar 2.4. bagian (b) menunjukkan *hyperplane* yang terbaik, yaitu yang terletak tepat pada tengah-tengah kedua *class*, sedangkan titik kotak dan lingkaran yang berada dalam lingkaran hitam adalah *support vector*. Upaya mencari lokasi *hyperplane* optimal ini merupakan inti dari proses pembelajaran pada SVM.

2.3.1. Karakteristik SVM

Berikut ini karakteristik yang dimiliki oleh algoritma Support Vector Machine (SVM):[7]

1. Secara prinsip SVM adalah *linear classifier*
2. *Pattern recognition* dilakukan dengan mentransformasikan data pada *input space* ke ruang yang berdimensi lebih tinggi, dan optimisasi dilakukan pada ruang vector yang baru tersebut. Hal ini membedakan SVM dari solusi *pattern recognition* pada umumnya, yang melakukan optimisasi parameter pada ruang hasil transformasi yang berdimensi lebih rendah daripada dimensi *input space*.
3. Menerapkan strategi *Structural Risk Minimization* (SRM)
4. Prinsip kerja SVM pada dasarnya hanya mampu menangani klasifikasi dua class.

[7] Erton F., "Support Vector Machine", Universitas KH. A. Wahab Hasbullah, 2018, hlm.

2.3.2. Kelebihan dan Kelemahan SVM

Dalam memilih solusi untuk menyelesaikan suatu masalah, kelebihan dan kelemahan masing-masing metode harus diperhatikan. Selanjutnya metode yang tepat dipilih dengan memperhatikan karakteristik data yang diolah. Dalam hal SVM, walaupun berbagai studi telah menunjukkan kelebihan metode SVM dibandingkan metode konvensional lain, SVM juga memiliki berbagai kelemahan.[8]

a. Kelebihan SVM

1) Generalisasi

Generalisasi didefinisikan sebagai kemampuan suatu metode untuk mengklasifikasikan suatu *pattern*, yang tidak termasuk data yang dipakai dalam fase pembelajaran metode itu. Vapnik menjelaskan bahwa generalization error dipengaruhi oleh dua faktor: error terhadap training set, dan satu faktor lagi yang dipengaruhi oleh dimensi VC (Vapnik-Chervokinensis). Strategi pembelajaran pada *neural network* dan umumnya metode *machine learning* difokuskan pada usaha untuk meminimalkan error pada training-set. Strategi ini disebut *Empirical Risk Minimization* (ERM). Adapun SVM selain meminimalkan error pada training-set, juga meminimalkan faktor kedua. Strategi ini disebut *Structural*

[8] Erton F., "Support Vector Machine", Universitas KH. A. Wahab Hasbullah, 2018, hlm.

Risk Minimization (SRM), dan dalam SVM diwujudkan dengan memilih hyperplane dengan margin terbesar. Berbagai studi empiris menunjukkan bahwa pendekatan SRM pada SVM memberikan error generalisasi yang lebih kecil daripada yang diperoleh dari strategi ERM pada neural network maupun metode yang lain.

2) *Curse of dimensionality*

Curse of dimensionality didefinisikan sebagai masalah yang dihadapi suatu metode *pattern recognition* dalam mengestimasikan parameter (misalnya jumlah hidden neuron pada neural network, stopping criteria dalam proses pembelajaran dsb.) dikarenakan jumlah sampel data yang relatif sedikit dibandingkan dimensional ruang vektor data tersebut. Semakin tinggi dimensi dari ruang vektor informasi yang diolah, membawa konsekuensi dibutuhkannya jumlah data dalam proses pembelajaran. Pada kenyataannya seringkali terjadi, data yang diolah berjumlah terbatas, dan untuk mengumpulkan data yang lebih banyak tidak mungkin dilakukan karena kendala biaya dan kesulitan teknis. Dalam kondisi tersebut, jika metode itu "terpaksa" harus bekerja pada data yang berjumlah relatif sedikit dibandingkan dimensinya, akan membuat proses estimasi parameter metode menjadi sangat sulit. *Curse of dimensionality* sering dialami

dalam aplikasi di bidang biomedical engineering, karena biasanya data biologi yang tersedia sangat terbatas, dan penyediaannya memerlukan biaya tinggi. Vapnik membuktikan bahwa tingkat generalisasi yang diperoleh oleh SVM tidak dipengaruhi oleh dimensi dari input vector. Hal ini merupakan alasan mengapa SVM merupakan salah satu metode yang tepat dipakai untuk memecahkan masalah berdimensi tinggi, dalam keterbatasan sampel data yang ada.

3) Landasan teori

Sebagai metode yang berbasis statistik, SVM memiliki landasan teori yang dapat dianalisa dengan jelas, dan tidak bersifat kuliah umum.

4) Feasibility

SVM dapat diimplementasikan relative mudah, karena proses penentuan support vector dapat dirumuskan dalam QP problem. Dengan demikian jika kita memiliki library untuk menyelesaikan QP problem, dengan sendirinya SVM dapat diimplementasikan dengan mudah. Selain itu dapat diselesaikan dengan metode sekuensial sebagaimana penjelasan sebelumnya.

b. Kelemahan SVM

SVM memiliki kelemahan atau keterbatasan, antara lain:

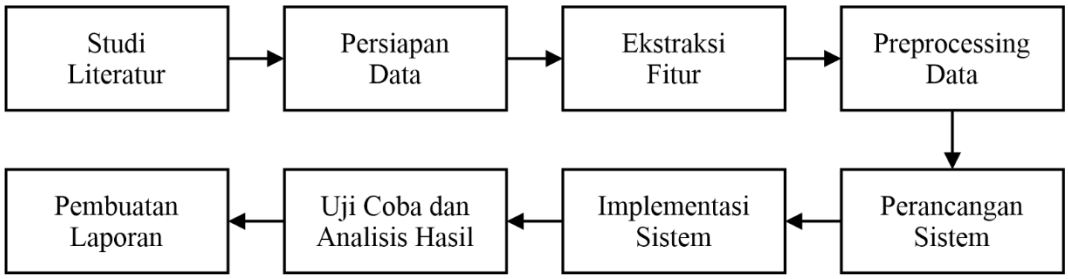
1) Sulit dipakai dalam *problem* berskala besar. Skala besar dalam hal ini dimaksudkan dengan jumlah sampel yang diolah.
2) SVM secara teoritik dikembangkan untuk *problem* klasifikasi dengan dua class. Dewasa ini SVM telah dimodifikasi agar dapat menyelesaikan masalah dengan *class* lebih dari dua, antara lain strategi *One versus rest* dan strategi *Tree Structure*. Namun demikian, masing-masing strategi ini memiliki kelemahan, sehingga dapat dikatakan penelitian dan pengembangan SVM pada *multiclass-problem* masih merupakan tema penelitian yang masih terbuka.

BAB III

METODOLOGI PENELITIAN

3.1. Tahapan Penelitian

Sistem yang diusulkan pada penelitian ini terdiri dari dua sistem utama yaitu sistem produksi dan sistem analisis yang di dalam sistem produksi terdapat proses penting yaitu ekstraksi fitur, sedangkan pada sistem analisis akan ada proses optimasi parameter dari algoritma klasifikasi yang akan digunakan. Algoritma yang akan digunakan yaitu *Support Vector Machine* (SVM) dengan *Decision Tree* dan *K-Nearest Neighbors* sebagai pembanding kinerja sistem. Adapun tahapan tahapan yang akan dilakukan pada penelitian ini dapat dilihat pada Gambar 3.1.



Gambar 3.1. Tahapan Penelitian

Berdasarkan diagram pada Gambar 3.1., tahapan penelitian secara garis besar dapat dijabarkan sebagai berikut:

1. Studi literatur merupakan tahap awal dalam penelitian ini. Tahap ini dilakukan untuk mengumpulkan penelitian terkait metode-metode yang digunakan baik dalam ekstraksi fitur, maupun klasifikasi. Pada tahap ini juga dilakukan pengumpulan landasan teori yang mendasari penelitian yang akan dilakukan.

2. Persiapan data merupakan tahapan yang dilakukan untuk mendapatkan dataset yang akan digunakan dalam melakukan klasifikasi *website*. Adapun dataset yang digunakan pada penelitian ini didapatkan dari *Kaggle Datasets*.
3. Ekstraksi fitur dilakukan untuk mengekstrak fitur-fitur yang terdapat pada sebuah *website* berdasarkan dengan dataset yang didapatkan dari *Kaggle Datasets*. Hasil ekstraksi fitur ini kemudian nantinya akan digunakan dalam melakukan deteksi *website phishing*.
4. *Preprocessing* data yang dilakukan berupa menganalisis data yang akan digunakan dan menyeleksi data berdasarkan hasil ekstraksi fitur yang telah dilakukan sebelumnya.
5. Perancangan sistem pada penelitian ini dilakukan untuk menentukan algoritma yang akan digunakan dalam klasifikasi *website phishing*. Dan adapun dalam penelitian ini, algoritma yang digunakan dalam melakukan klasifikasi adalah algoritma *Support Vector Machine* (SVM) dengan *Decision Tree* dan *K-Nearest Neighbors* sebagai pembanding kinerja sistem. Pada tahap ini juga dilakukan pembuatan *flowchart* terkait alur kerja sistem.
6. Implementasi sistem dilakukan sesuai dengan *flowchart* yang telah dibuat pada tahap sebelumnya. Pada penelitian ini, sistem dibuat dengan menggunakan bahasa pemrograman python.
7. Uji coba sistem dilakukan untuk mengetahui seberapa akurat sistem yang dibuat. Uji coba nantinya akan dilakukan dengan beberapa skenario yang beda-beda.

8. Tahap akhir dalam penelitian ini adalah melakukan penulisan laporan penelitian dalam bentuk skripsi sebagai bahan publikasi.

3.2. Waktu dan Tempat Penelitian

Penelitian ini dilakukan selama 9 bulan yang dimulai sejak disetujuinya proposal penelitian ini pada bulan Maret 2019 hingga proses pelaporan hasil penelitian ini pada bulan November 2019. Penelitian ini dilakukan di *Ubiquitous Computing and Networking Laboratorium* (UBICON) Departemen Teknik Informatika, Fakultas Teknik, Universitas Hasanuddin.

3.3. Instrumen Penelitian

Instrumen yang digunakan pada penelitian ini adalah:

1. Software
    a. Sistem operasi: Linux Ubuntu 18.04 LTS
    b. Bahasa Pemrograman: Python 3.6
    c. Text Editor: Sublime 3 Build 2112
2. Hardware
    a. Laptop HP 14-bw0xx, RAM 4GB, Processor AMD A9-9420 Radeon R5, 5 Compute Cores 2C + 3G (2 CPUs), 3.0GHz

3.4. Ekstraksi Fitur

Ekstraksi fitur pada penelitian ini dilakukan untuk mengekstrak fitur-fitur yang terdapat pada sebuah *website* dengan cara memasukkan URL ke dalam sistem kemudian sistem akan mengakses URL tersebut dan melakukan ekstraksi fitur pada *website* tersebut. Karena untuk melakukan ekstraksi harus mengakses *website*

yang akan diprediksi maka sistem ini harus terhubung ke internet untuk melakukan hal tersebut. Jadi otomatis sistem ini nantinya merupakan sistem yang online.

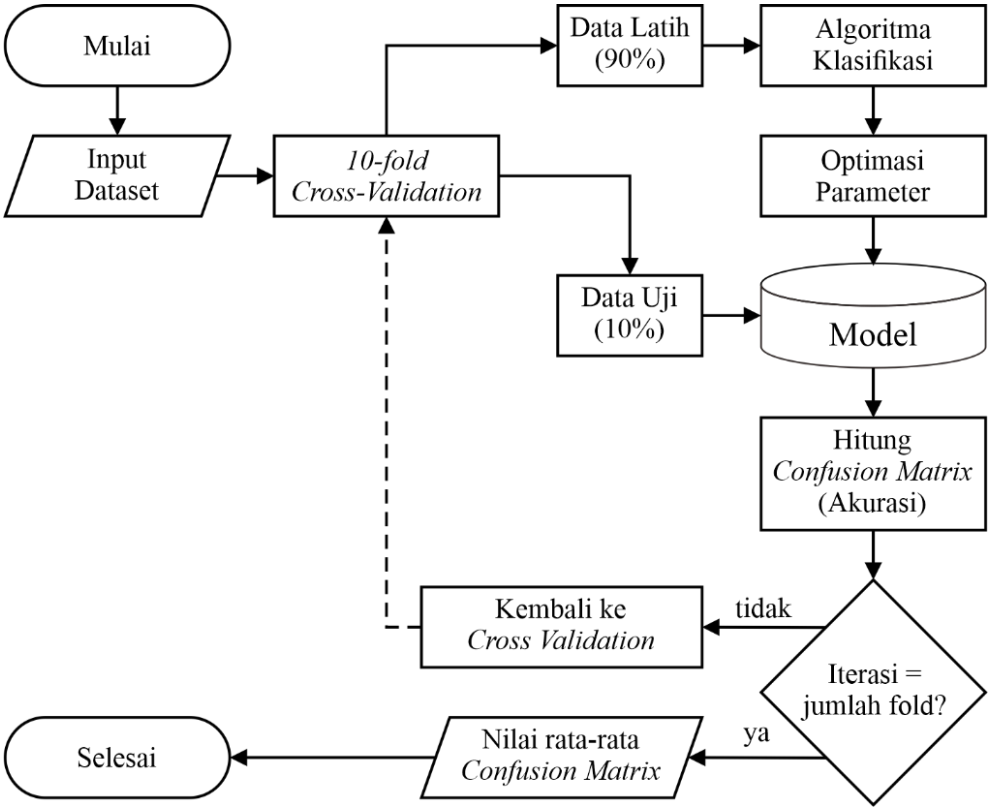
Adapun metode yang digunakan dalam melakukan ekstraksi fitur ini yaitu diantaranya melakukan pengecekan pada URL yang dalam hal ini tidak perlu langsung mengakses *website* tersebut melainkan cukup melakukan operasi pada URL yang dimasukkan. Kemudian ada metode lain yaitu melakukan pengecekan pada source page yang artinya sistem harus dapat mengakses *website* tersebut secara online. Metode lain diantaranya yaitu melakukan pengecekan pada data *Whois, DNS Record, Alexa database, website PhishTank* dan *StopBadware*.

3.5. Perancangan dan Implementasi Sistem

Sistem yang dibuat terdiri atas dua bagian, yaitu sistem implementasi dan sistem evaluasi. Sistem implementasi merupakan sistem yang akan dapat digunakan langsung oleh *user* dengan memasukkan input berupa URL. Sedangkan sistem evaluasi adalah sistem yang dibuat untuk menganalisis kinerja sistem implementasi.

3.5.1. Sistem Evaluasi

Sistem evaluasi ini merupakan sistem yang dirancang sebagai sistem evaluasi kinerja sistem produksi. Evaluasi pada sistem ini dilakukan dengan cara mengevaluasi dataset yang digunakan pada sistem implementasi. *Flowchart* sistem evaluasi dapat dilihat pada Gambar 3.2.



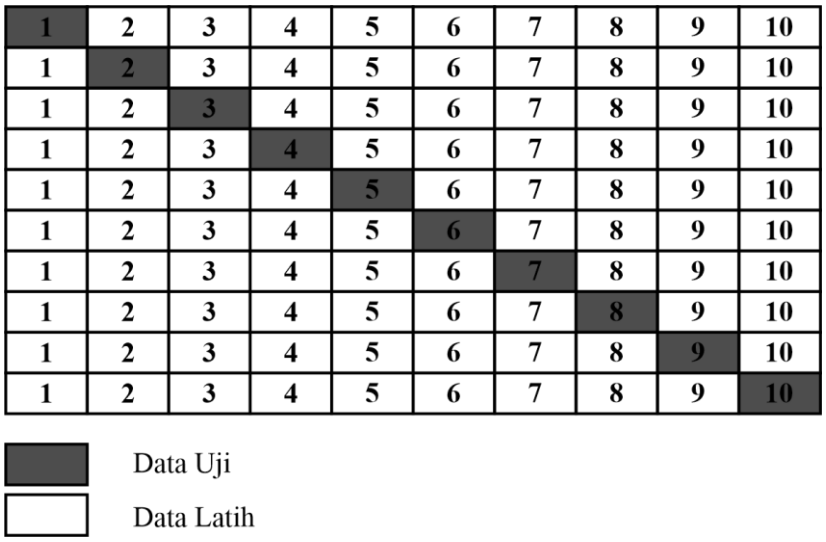
Gambar 3.2. Flowchart Sistem Evaluasi

1. Input Dataset

   Dataset yang digunakan pada sistem evaluasi ini merupakan dataset yang sama dengan yang digunakan pada sistem produksi, hanya saja yang membedakanny adalah dataset pada sistem produksi ini menggunakan *cross-validation.*

   Cross-validation merupakan metode statistika untuk mengevaluasi dan membandingkan algoritma pembelajaran dengan membagi data menjadi dua bagian. Satu bagian untuk melatih model dan bagian lainnya untuk memvalidasi model

tersebut. Salah satu bentuk *cross-validation* adalah *k-fold cross-validation.*



Gambar 3.3. *10-Folf Cross-validation*

K-fold cross-validation merupakan metode yang digunakan untuk membagi dataset menjadi data latih dan data uji. *K-fold cross-validation* akan membagi menjadi *k* bagian dengan ukuran yang sama. Nilai *k* yang digunakan pada penelitian ini adalah 10. *10-fold cross-validation* adalah salah satu *k-fold cross-validation* yang direkomendasikan untuk pemilihan model terbaik karena cenderung memberikan estimasi akurasi yang kurang bias dibandingkan dengan *cross-validation* biasa, *leave-one-out cross-validation* dan *bootstrap*. Dalam *10-fold cross-validation*, data dibagi menjadi 10 bagian berukuran kira-kira sama, sehingga kita memiliki 10 bagian data untuk mengevaluasi kinerja model atau algoritma. Untuk masing-masing dari 10 bagian data

tersebut, *cross-validation* akan menggunakan 9 bagian untuk pelatihan dan 1 bagian untuk pengujian seperti diilustrasikan pada Gambar 3.3.

2. Algoritma Klasifikasi

Algoritma klasifikasi pada sistem ini berperan dalam pembuatan model untuk data latih yang kemudian nantinya akan digunakan pada sistem dalam melakukan proses klasifikasi. Pada tahap ini proses klasifikasi akan dapat melakukan prediksi apakah data tersebut bernilai 1 yang berarti *phishing* ataukah bernilai -1 yang berarti *legitimate*. Pada sistem ini digunakan beberapa algoritma klasifikasi yaitu *Support Vector Machine (SVM)*, *K-Nearest Neighbour*, dan *Decision Tree*. Jadi dengan menggunakan tiga algoritma tersebut maka *user* dapat memilih algoritma mana yang akan digunakan.

a. *Support Vector Machine* (SVM)

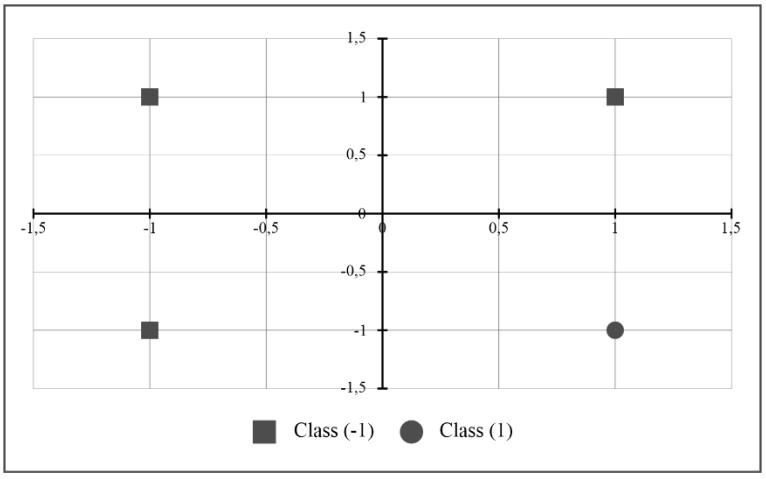
Support Vector Machine (SVM) merupakan algoritma yang saat ini disebut-sebut mendapatkan banyak perhatian dikarenakan kelebihan yang dimilikinya dibandingkan dengan algoritma yang lain. SVM adalah algoritma yang bekerja dengan mencoba menemukan *hyperplane* terbaik dalam memisahkan setiap *class.*

Berikut penjelasan singkat proses kerja sistem dengan menggunakan algoritma klasifikasi SVM dengan dua fitur dataset:

Tabel 3.1. Dataset dengan 4 data dan 2 Fitur

No.	Age of Domain (x_1)	Web Traffic (x_2)	Result
1	-1	-1	-1
2	1	1	-1
3	-1	1	-1
4	1	-1	1

Hasil visualisasi data dapat dilihat pada Gambar 3.4:



Gambar 3.4. Visualisasi Data

Karena ada dua fitur (x_1 dan x_2), maka w juga akan memiliki 2 fitur (w_1 dan w_2). Persamaan yang digunakan adalah sebagai berikut:

$$\frac{1}{2}\|w\|^2 = \frac{1}{2}(w_1^2 + w_2^2) \qquad (3.1)$$

Ket:

w = vector fitur yang dimiliki oleh data

Dengan syarat:

$y_i(w.x_i + b) \geq 1, \quad i = 1,2,3, \ldots, N$

$y_i(w_1.x_1 + w_2.x_2 + b) \geq 1$

Ket:

x_i = fitur yang dimiliki oleh data

y_i = kelas data

w = vector fitur

b = jumlah bias

Sehingga didapatkan persamaan sebagai berikut:

$$1.\ (w_1 + w_2 - b) \geq 1, \quad untuk\ y_1 = -1, x_1 = -1, x_2 = -1 \tag{3.2}$$

$$2.\ (-w_1 - w_2 - b) \geq 1, \quad untuk\ y_2 = -1, x_1 = 1, x_2 = 1 \tag{3.3}$$

$$3.\ (w_1 - w_2 - b) \geq 1, \quad untuk\ y_3 = -1, x_1 = -1, x_2 = 1 \tag{3.4}$$

$$4.\ (w_1 - w_2 + b) \geq 1, \quad untuk\ y_4 = 1, x_1 = 1, x_2 = -1 \tag{3.5}$$

- Menjumlahkan persamaan (1) dan (2):

$$\begin{array}{l} (w_1 + w_2 - b) \geq 1 \\ (-w_1 - w_2 - b) \geq 1 \\ \hline -2b = 2 \end{array} +$$

Maka b = -1

- Menjumlahkan persamaan (1) dan (4):

$$\begin{array}{l} (w_1 + w_2 - b) \geq 1 \\ (-w_1 - w_2 - b) \geq 1 \\ \hline 2w_1 = 2 \end{array} +$$

Maka $w_1 = 1$

- Menjumlahkan persamaan (2) dan (4):

$$\begin{array}{l} (w_1 + w_2 - b) \geq 1 \\ \underline{(-w_1 - w_2 - b) \geq 1} \\ -2w_2 = 2 \end{array} +$$

Maka $w_2 = -1$

Sehingga didapatkan persamaan *hyperplane:*

$$w_1 x_1 + w_2 x_2 + b = 0 \quad (3.6)$$

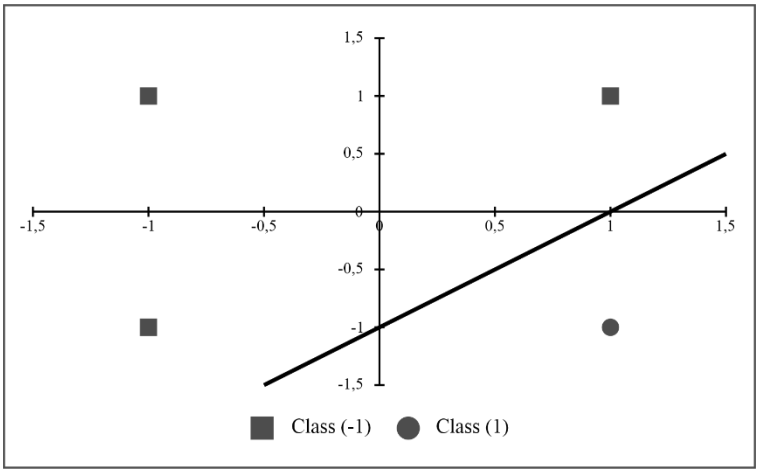
$$x_1 + x_2 - 1 = 0 \quad (3.7)$$

$$x_2 = 1 - x_1 \quad (3.8)$$

Visualisasi garis *hyperplane* sebagai fungsi klasifikasi:

Tabel 3.2. Pengujian nilai x_1

x_1	$x_2 = 1 - x_1$
-2	3
-1	2
0	1
1	0
2	-1



Gambar 3.5. Hasil Penentuan Garis *Hyperplane*

Setelah didapatkan *hyperplane* terbaik maka selanjutnya adalah memasukkan data uji untuk dilakukan proses klasifikasi. Misalnya dengan nilai $x_1 = -1$ dan $x_2 = 1$.

$$f(x) = x_1 + x_2 - 1 \tag{3.9}$$

Kelas = $sign(f(x))$

Tabel 3.3. Hasil Klasifikasi Data Uji

Data Uji		Hasil Klasifikasi
x_1	x_2	Kelas = $sign(x_1 + x_2 - 1)$
-1	1	$sign(-1 + 1 - 1) = -1$

Sebenarnya yang membuat SVM menjadi metode yang lebih unggul dibandingkan yang lain adalah pada *kernel trick* yang dimilikinya. Dengan *kernel trick* SVM dapat memisahkan setiap *class* dalam dimensi yang lebih tinggi sehingga dapat menemukan *hyperplane* terbaik. Fungsi

kernel yang digunakan pada penelitian ini adalah sebagai berikut.

1) *Kernel Linear*

$$K(\mathrm{x},\mathrm{y}) = \mathrm{x}.\mathrm{y} \tag{3.10}$$

2) *Kernel Polynomial*

$$K(\mathrm{x},\mathrm{y}) = (\mathrm{x}.\mathrm{y})^{d} \tag{3.11}$$

3) *Kernel RBF*

$$K(\mathrm{x},\mathrm{y}) = \exp\left(\frac{-\|x-y\|^2}{2\sigma^2}\right) \tag{3.12}$$

b. *Decision Tree*

Decision tree atau yang biasa disebut pohon keputusan adalah salah satu metode klasifikasi yang paling populer, karena mudah untuk diinterpretasi oleh manusia. *Decision tree* adalah model prediksi menggunakan struktur pohon atau struktur berhirarki. *Decision Tree* adalah pohon dimana setiap cabangnya menunjukkan pilihan diantara sejumlah alternatif pilihan yang ada, dan setiap daunnya menunjukkan keputusan yang dipilih. *Decision tree* biasa digunakan untuk mendapatkan informasi untuk tujuan pengambilan sebuah keputusan. *Decision tree* dimulai dengan sebuah *root node* (titik awal) yang dipakai oleh *user*

untuk mengambil tindakan. Dari *root node* ini, *user* melakukan *split* (pemecahan) sesuai dengan algoritma *decision tree*. Hasil akhirnya adalah sebuah *decision tree* dengan setiap cabangnya menunjukkan kemungkinan skenario dari keputusan yang diambil serta hasilnya.

Algoritma *decision tree* dapat dimaksimalkan hasil keputusannya dengan mengoptimasi beberapa parameternya antara lain *criterion* yang berfungsi dalam proses *split* dan parameter *max_depth* yang berfungsi untuk mengatur kedalam pohon keputusan yang akan dibuat. *Criterion* memiliki dua nilai yaitu *gini* dan *entropy*. Rumus setiap nilai *criterion* dapat dilihat pada persamaan 3.13. dan 3.14.

$$\text{Gini} = \sum_{j=1}^{k} -p_j^2 \tag{3.13}$$

$$\text{Entropy} = \sum_{j=1}^{k} -p_j \log_2 p_j \tag{3.14}$$

Ket:

k = banyaknya partisi pada himpunan (dataset) kasus

p_j = probabilitas yang di dapat dari Sum(Ya) dibagi Total Kasus.

Berikut penjelasan singkat proses kerja sistem dengan menggunakan algoritma klasifikasi Decision Tree dengan dua fitur dataset:

Tabel 3.4. Dataset dengan 4 Data Berlabel dan 2 Fitur

DNS Record (X1)	Web Traffic (X2)	Result
-1	-1	-1
-1	1	-1
-1	0	1
1	1	1
-1	-1	?

$$\text{Gain(A)} = \text{Entropi(S)} - \sum_{i=1}^{k} \frac{|S_i|}{|S|} \times Entropi(S_i) \quad (3.15)$$

Ket:

S = ruang (data) sample yang digunakan untuk training.

A = atribut.

$|S_i|$ = jumlah sample untuk nilai V.

$|S|$ = jumlah seluruh sample data.

Entropi(S_i) = entropy untuk sampel-sampel yang memiliki nilai *i*.

Data yang telah ada pada Tabel 3. akan digunakan untuk membentuk pohon keputusan dimana memiliku fitur-fitur seperti *Having Sub Domain*, *Age of Domain*, *DNS Record*, dan *Web Traffic*. Setiap fitur memiliki nilai. Sedangkan

kelasnya ada pada kolom *Result* yaitu kelas (-1) dan kelas (1). Dataset tersebut memiliki empat kasus yang terdiri dua kelas (-1) dan dua kelas (1).

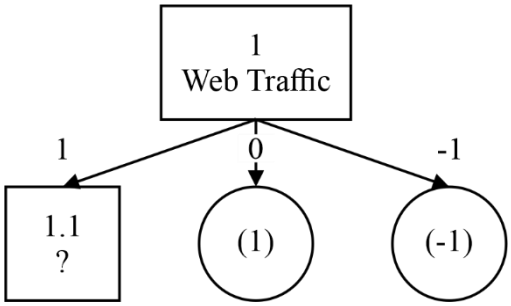
$$Entropy\ (S) = \left(\frac{-2}{4} \times \log_2\left(\frac{2}{4}\right)\right) + \left(\frac{-2}{4} \times \log_2\left(\frac{2}{4}\right)\right) = 1$$

Setelah mendapatkan total *entropy*, selanjutnya dilakukan analisis pada setiap fitur dan nilai-nilainya dan hitung entropinya.

Tabel 3.5. Hasil Perhitungan Total Entropy

Node	Fitur	Nilai	Sum (Nilai)	Sum (-1)	Sum (1)	Entropy	Gain
1	DNS Record	-1	3	2	1	0,9234	
		1	1	0	1	0	
							0,3074
	Web Traffic	-1	1	1	0	0	
		0	1	0	1	0	
		1	2	1	1	1	
							0,5

Dari tabel di atas didapatkan nilai *gain* terbesar yaitu pada fitur *Web Traffic*. Oleh karena itu, fitur *web traffic* akan menjadi *root node* (node akar). Sehingga menghasilkan gambar pohon seperti pada gambar berikut ini.



Gambar 3.6. Pohon Node 1 (root node)

Tabel 3.6. Hasil Filter Fitur Web Traffic dengan Nilai (1)

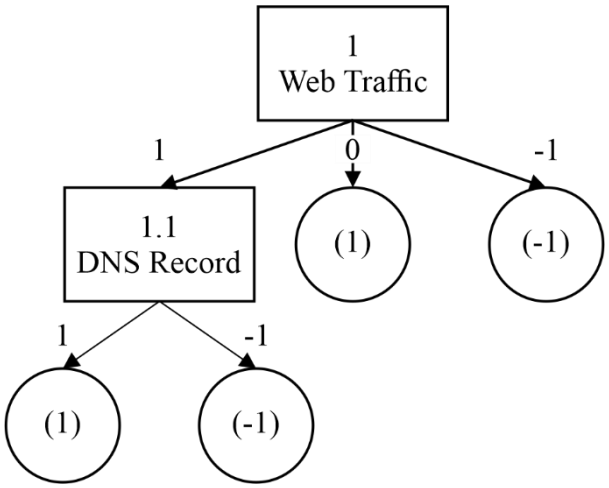
DNS Record (X1)	Web Traffic (X2)	Result
-1	1	-1
1	1	1

Setelah difilter kemudian hitung nilai entropi fitur "*Web Traffic*" dengan nilai (1) dan entropi setiap fitur serta gainnya. Setelah itu tentukan fitur yang memiliki gain tertinggi untuk dibuatkan node berikutnya.

$$Entropy\ (S) = \left(\frac{-1}{2} \times \log_2\left(\frac{1}{2}\right)\right) + \left(\frac{-1}{2} \times \log_2\left(\frac{1}{2}\right)\right) = 1$$

Tabel 3.7. Hasil Analisis Akhir

Node	Fitur	Nilai	Sum (Nilai)	Sum (-1)	Sum (1)	Entropy	Gain
1.1	DNS Record	-1	1	1	0	0	
		1	1	0	1	0	
							1



Gambar 3.7. Pohon Keputusan Akhir

c. *K-Nearest Neighbour*

K-Nearest Neighbors (KNN) merupakan algoritma yang mengelompokkan data ke dalam kelompok yang memiliki sifat termirip atau tetangga yang terdekat darinya. KNN mencari *K feature vector* dengan sifat termirip, kemudian mengelompokkan data baru ke dalam kelompok *feature vector* tersebut.

Berikut penjelasan singkat proses kerja sistem dengan menggunakan algoritma klasifikasi KNN dengan empat fitur dataset:

Tabel 3.8. Dataset dengan 4 Data Berlabel dan 4 Fitur

Having Sub Domain (X1)	Age of Domain (X2)	DNS Record (X3)	Web Traffic (X4)	Result
-1	-1	-1	-1	-1
-1	-1	-1	1	-1
1	-1	-1	0	1
1	1	1	1	1
-1	1	-1	-1	?

Ditentukan Nilai $K = 3$.

Persamaan *Euclidean Sistance:*

$$[(x, y), (a, b)] = \sqrt{(x - a)^2 + (y - b)^2} \qquad (3.16)$$

Ket:

(x, y) = fitur data yang berlabel

(a, b) = fitur data baru yang belum berlabel

Menghitung jarak antara data baru dengan semua data latih menggunakan persamaan *Euclidean Distance*.

Tabel 3.9. Hasil Perhitungan *Euclidean Distance*

X1	X2	X3	X4	*Euclidean Distance* (-1, 1, -1, -1)
-1	-1	-1	-1	2
-1	-1	-1	1	2,82
1	-1	-1	0	3
1	1	1	1	3,46

Setelah didapatkan nilai *Euclidean Distance*, data kemudian diurutkan dan ditentukan tiga tetangga terdekat seperti yang telah ditentukan sebelumnya yaitu K=3.

Tabel 3.10. Hasil Pengurutan dan Penentuan Tetangga Terdekat

X1	X2	X3	X4	*Euclidean Distance*	Urutan Jarak	3-NN?	Kategori Ya untuk KNN
1	1	1	1	3,46	4	Tidak	-
1	-1	-1	0	3	3	Ya	1
-1	-1	-1	1	2,82	2	Ya	-1
-1	-1	-1	-1	2	1	Ya	-1

Dari hasil pengurutan nilai *Euclidean Distance* dan penentuan tetangga terdekat di atas dapat dilihat bahwa terdapat satu data dengan kelas (1) dan dua data dengan kelas (-1). Sehingga dari data tersebut dapat diklasifikasikan bahwa data uji yang dimasukkan merupakan data dengan kelas (-1).

3. Optimasi Parameter

Optimasi parameter merupakan metode yang digunakan untuk mengoptimalkan parameter pada sebuah algoritma sehingga dapat mendapatkan hasil yang lebih baik daripada parameter defaultnya. Pada penelitian ini optimasi parameter dilakukan dengan cara mengubah-ubah nilai dari parameter yang terdapat pada sebuah algoritma.

a. *Support Vector Machine* (SVM)

Algoritma SVM memiliki tiga *kernel* utama yaitu *Linear*, *Polynomial*, dan *Gaussian radial basis function* (RBF). Optimasi parameter akan dilakukan terhadap tiga *kernel* dengan cara mengubah-ubah nilai parameter yang dimiliki oleh masing-masing *kernel.*

1) *Kernel Linear*

Pada *kernel linear*, optimasi parameter hanya dilakukan dengan mengubah-ubah nilai satu parameter yaitu parameter *C.* Parameter *C* ini merupakan besaran nilai penalty yang diberikan terhadap error klasifikasi. Nilai parameter *C* yang digunakan yaitu 1.0, 1.5, 2.5, dan 5.0.

2) *Kernel Polynomial*

Terdapat dua parameter utama yang dilakukan optimasi pada *kernel polynomial* yaitu parameter

degree dan *C*. Parameter *C* merupakan besaran nilai penalty yang diberikan terhadap error klasifikasi. Pada parameter *degree* akan digunakan nilai 1, 3, 5, 7, dan 9. Sedangkan *C* akan digunakan nilai 1.0, 1.5, 2.5, dan 5.0.

3) *Kernel RBF*

Pada *kernel RBF* dilakukan optimasi parameter terhadapat dua parameter utama yaitu paramaeter *gamma* dan *C*. Parameter *C* merupakan besaran nilai penalty yang diberikan terhadap error klasifikasi. Pada parameter *gamma* akan digunakan nilai 0.1, 0.5, 1.0, 1.5 dan 2.0. Sedangkan pada parameter *C* digunakan nilai 1.0, 1.5, 2.5, dan 5.0.

b. *Decision Tree*

Optimasi parameter yang dilakukan pada algoritma *decision tree* adalah dengan mengubah-ubah parameter *criterion* dan *max_depth*. Parameter *criterion* berfungsi pada proses *split*, apakah menggunakan *gini* atau *entropy*. Sedangkan *max_depth* adalah kedalaman pohon yang akan dibuat. Pada parameter *criterion* hanya terdapat dua nilai yaitu *gini* dan *entropy*. Sedangkan pada parameter *max_depth* akan digunakan nilai 1, 5, 10, 15, 20.

c. *K-Nearest Neighbors*

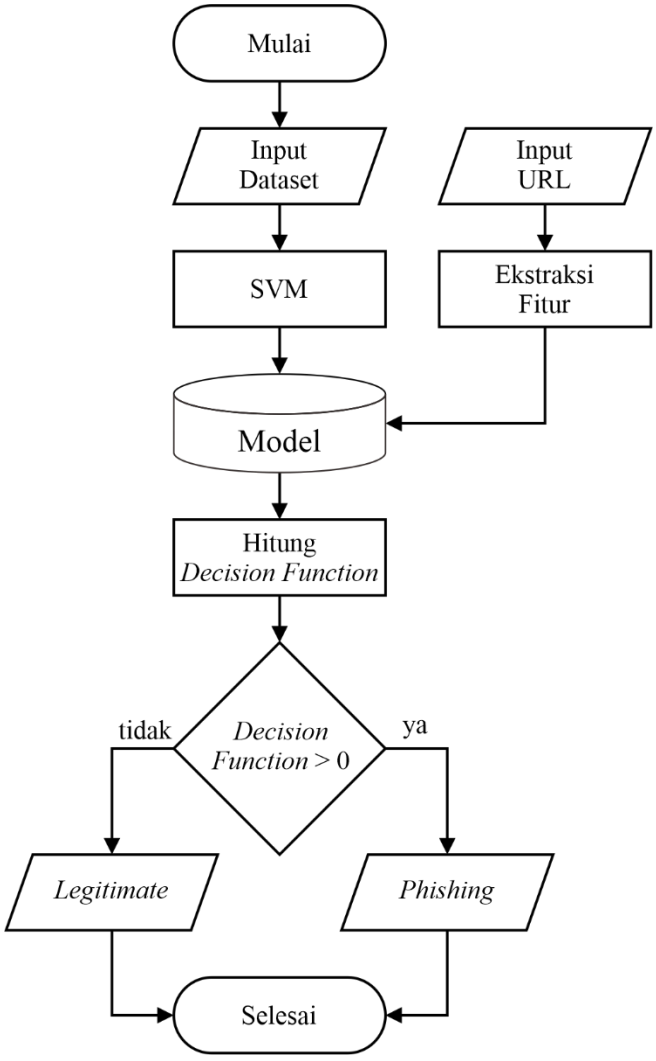
Optimasi parameter yang dilakukan pada algoritma *k-nearest neighbors* adalah dengan mengubah-ubah parameter *n-neighbors* dan *weights*. Parameter *n_neighbors* merupakan parameter yang berfungsi untuk menentukan jumlah *neighbors* (tetangga) terdekat yang akan diambil untuk menentukan *class* titik poin yang akan diprediksi. Sedangkan parameter *weights* berfungsi untuk memberikan bobot pada setiap *neighbors*. Parameter *weights* berisikan dua nilai yaitu *uniform* dan *distance*. *Uniform* berarti semua *neighbors* diberikan bobot yang sama. Sedangkan *distance* berarti bobot akan ditentukan berdasarkan jaraknya dari titik poin, ini artinya bahwa *neighbors* terdekat akan memiliki pengaruh yang lebih besar dibandingkan dengan *neighbors* yang jaraknya lebih jauh. Pada parameter *weights* akan digunakan dua nilai yaitu *uniform* dan *distance*. Sedangkan pada parameter *n-neighbors* akan digunakan nilai 1, 5, 10, 15, 20.

4. Akurasi Klasifikasi

Akurasi klasifikasi merupakan hasil perhitungan nilai rata-rata akurasi *confusion matrix*. Hasil yang dikeluarkan akan berupa persentase akurasi dari setiap algoritma yang digunakan yaitu *support vector machine* (SVM) dengan tiga kernelnya (*linear*,

polynomial, dan *RBF*), *decision tree*, dan *k-nearest neighbors*. Dengan akurasi dari masing-masing algoritma serta dengan optimasi parameternya akan ditemukan algoritma mana yang memiliki akurasi terbaik serta optimasi parameternya.

3.5.2. Sistem Implementasi



Gambar 3.8. Flowchart Sistem Implementasi

Sistem produksi ini merupakan sistem yang dirancang untuk digunakan diimplementasikan dalam melakukan deteksi *phishing*. Hal yang

membedakan sistem produksi ini dengan sistem evaluasi adalah sistem produksi dirancang untuk digunakan oleh *user* dengan masukan berupa URL. Selain perbedaan tersebut ada proses ekstraksi fitur yang sangat penting dan menentukan kinerja sistem. Untuk lebih jelasnya flowchart sistem dapat dilihat pada Gambar 3.8.

Flowchart di atas merupakan gambaran proses kerja pada sistem produksi sehingga dapat melakukan klasisfikasi *website phishing*. Berdasarkan flowchart tersebut, sistem produksi secara garis besar dapat dijabarkan sebagai berikut:

1. Input Dataset

   Langkah pertama adalah input dataset, pada tahap ini dipersiapkan data *website phishing* yang akan menjadi data latih untuk sistem. Pada penelitian ini, dataset didapatkan dari *Kaggle Dataset*. Dataset yang digunakan ini merupakan dataset yang telah biasa digunakan dalam penelitian-penelitian untuk melakukan deteksi *website phishing*. Data yang digunakan dalam pembuatan sistem berjumlah 11055 data *website phishing* yang terdiri dari 4898 *website* '*legitimate*' dan 6157 *website* '*phishing*' dengan 17 fitur. Gambaran dataset dapat dilihat pada Tabel 3.11.

Tabel 3.11. Gambaran Dataset *Website Phishing*

Having IP Address	URL Length	Shortining Service	Having At Symbol	Double Slash Redirecting	Prefix Suffix	Having Sub Domain	Domain Registration Length	HTTPS Token	Submitting to Email	Right Click	Iframe	Age of Domain	DNS Record	Web Traffic	Statistical Report	Result
-1	1	1	1	-1	-1	-1	-1	-1	-1	1	1	-1	-1	-1	-1	-1
1	1	1	1	1	-1	0	-1	-1	1	1	1	-1	-1	0	1	-1
1	0	1	1	1	-1	-1	-1	-1	-1	1	1	1	-1	1	-1	-1
1	0	1	1	1	-1	-1	1	-1	1	1	1	-1	-1	1	1	-1
1	0	-1	1	1	-1	1	-1	1	1	1	1	-1	-1	0	1	1
-1	0	-1	1	-1	-1	1	-1	-1	-1	1	1	1	1	1	-1	1
1	0	-1	1	1	-1	-1	1	1	-1	1	1	1	-1	-1	-1	-1
1	0	1	1	1	-1	-1	1	-1	1	1	1	-1	-1	0	1	-1
1	0	-1	1	1	-1	1	-1	-1	1	1	1	1	-1	1	1	1
1	1	-1	1	1	-1	-1	-1	1	1	1	1	1	-1	0	1	-1
1	1	1	1	1	-1	0	1	1	-1	1	1	-1	1	1	-1	1
1	1	-1	1	1	-1	1	-1	1	-1	1	1	-1	-1	-1	-1	-1
-1	1	-1	1	-1	-1	0	1	-1	1	1	1	1	-1	-1	1	-1
1	1	-1	1	1	-1	0	1	1	1	1	1	-1	-1	0	1	-1
1	1	-1	1	1	1	-1	-1	-1	1	1	1	1	-1	1	1	1
1	-1	-1	-1	1	-1	0	1	1	1	1	1	1	-1	-1	1	-1
1	-1	-1	1	1	-1	1	-1	-1	-1	1	1	1	-1	0	-1	-1
1	-1	1	1	1	-1	-1	1	1	-1	1	1	-1	1	1	-1	-1
1	1	1	1	1	-1	-1	1	-1	-1	1	1	1	-1	-1	-1	1
1	1	1	1	1	-1	-1	-1	1	-1	-1	-1	1	-1	0	-1	1
1	0	-1	1	1	-1	0	-1	1	-1	1	1	-1	1	1	-1	1
1	0	1	1	1	-1	0	1	-1	-1	1	1	-1	1	-1	-1	1

Adapun penjelasan untuk setiap fitur yang digunakan pada penelitian ini dapat di lihat pada tabel di bawah ini:

Tabel 3.12. Penjelasan Fitur Dataset *Website Phishing*

NO.	FITUR	VALUE	KETERANGAN
1	Having IP Address	1, -1	Adanya IP address sebagai nama domain

			pada URL (biner) -1 (tidak), 1 (iya)
2	URL Length	1, 0, -1	Panjang URL (polinomial) -1 (kurang dari 54 karakter), 0 (antara 54 sampai 75 karakter), 1 (lebih dari 75 karakter)
3	Shortening Service	1, -1	Penggunaan layanan penyingkat URL (biner) -1 (tidak), 1 (iya)
4	Having At Symbol	1, -1	Penggunaan simbol "@" pada URL (biner) -1 (tidak), 1 (iya)
5	Double Slash Redirecting	1, -1	Penggunaan simbol "//" pada URL untuk mengalihkan website (biner) -1 (tidak), 1 (iya)
6	Prefix Suffix	1, -1	Penggunaan simbol "-" pada nama domain URL (biner) -1 (tidak), 1 (iya)
7	Having Sub Domain	1, 0, -1	Penggunaan sub domain (polinomial) -1 (tidak punya), 0 (1 sub domain), 1 (lebih dari sub domain)
8	Domain Registration Length	1, -1	Batas berlakunya domain (biner) -1 (lebih dari 1 tahun), 1 (kurang dari 1 tahun)
9	HTTPS Token	1, -1	Penggunaan "HTTPS" ke dalam bagian domain pada URL (biner) -1 (tidak), 1 (iya)
10	Submitting to Email	1, -1	Penggunaan fungsi "mail() atau mailto" dalam php untuk mengirim informasi user (biner) -1 (tidak), 1 (iya)

11	Right Click	1, -1	Keadaan klik kanan pada website (biner) -1 (diaktifkan), 1 (dinonaktifkan)
12	Ifram	1, -1	Penggunaan fungsi iframe (biner) -1 (tidak), 1 (iya)
13	Age of Domain	1, -1	Umur domain (biner) -1 (lebih dari atau sama dengan 6 bulan), 1 (kurang dari 6 bulan)
14	DNS Record	1, -1	Adanya catatan DNS pada domain (biner) -1 (ada), 1 (tidak ada)
15	Web Traffic	1, 0, -1	Rank web traffix dalam basis data Alexa (polinomial) -1 (di atas 100.000), 0 (di bawah 100.000), 1 (tidak terdaftar)
16	Statistical Report	1, -1	Host berasal dari Top Phishing IPs atau Top Phishing Domains yang dibuat oleh pihak StopBadware dan PhishTank (biner) -1 (tidak), 1(iya)

2. Input URL

Pada langkah dataset sebelumnya telah di ketahui bahwa dataset yang telah didapatkan dari Kaggle Dataset akan digunakan sebagai data latih sedangkan yang akan menjadi data uji adalah masukan URL dari sebuah *website*, yang kemudian akan di ekstrak berdasarkan fitur-fitur yang ada pada dataset yang digunakan. Masukan URL disini sebagai data uji sekaligus menjadi data yang diklasifikasikan oleh sistem, apakah *website* tersebut adalah *phishing* atau bukan *phishing*.

3. Ektraksi Fitur

Setelah URL dimasukkan maka langkah selanjutnya adalah melakukan ekstraksi fitur pada *website* berdasarkan URL yang telah dimasukkan. Adapun fitur-fitur yang diekstrak adalah sebagai berikut.

a. Having IP Address

Fitur ini merupakan fitur yang mengecek apakah URL pada *website* menggunakan *IP address* sebagai hostname atau tidak, misalnya pada contoh URL berikut ini:

"http://125.98.3.123/fake.html"

Untuk melakukan pengecekan tersebut dengan menggunakan python maka yang harus dilakukan adalah melakukan parsing pada URL dengan mengecek pada bagian hostname, apakah URL tersebut menggunakan IP address sebagai hostname atau tidak. Apabila URL tersebut tidak menggunakan IP address sebagai hostname maka sistem memberikan nilai -1, sedangkan apabila URL tersebut menggunakan IP address sebagai hostname maka sistem akan memberikan nilai 1.

Tabel 3.13. Nilai Fitur *Having IP Address*

Legitimate	*Phishing*
-1	1

b. URL Length

Fitur ini merupakan fitur yang mengecek panjang URL yang digunakan oleh sebuah *website*, misalnya URL di bawah ini yang memiliki URL yang terlalu panjang:

“http://federmacedoadv.com.br/3f/aze/ab51e2e319e51502f416dbe46b773a5e/?cmd=_home&dispatch=11004d58f5b74f8dc1e7c2e8dd4105e811004d58f5b74f8dc1e7c2e8dd4105e8@phishing.website.html”

Untuk melakukan ekstraksi fitur ini maka yang dilakukan adalah dengan menghitung jumlah karakter yang digunakan oleh URL tersebut. Apabilah URL tersebut memiliki panjang URL kurang dari 54 karakter maka sistem memberikan nilai -1, sedangkan apabila panjang URL kurang dari 75 karakter maka sistem memberikan nilai 0, dan apabila panjang URL di atas 75 karakter maka sistem memberikan nilai 1.

Tabel 3.14. Nilai Fitur *URL Length*

Legitimate	*Suspicious*	*Phishing*
-1	0	1

c. Shortening Service

URL shortening service merupakan sebuah metode yang dapat digunakan menyingkat URL yang panjang menjadi lebih pendek seperti “http://portal.hud.ac.uk/” yang dapat disingkat menjadi “bit.ly/19DXSk4”. Ekstraksi fitur ini dilakukan dengan cara melakukan pengecekan pada URL tersebut dengan menggunakan 10 URL shortening service.

Tabel 3.15. Daftar Shortener Service

NO.	Shortener Service	Bentuk Singkatan
1	BItly	bit.ly
2	Google	goo.gl
3	TinyURL	tinyurl.com
4	Buffer	buff.ly
5	Adf.ly	adf.ly
6	Hootsuite	ow.ly
7	Polr	polr.me
8	Is.gd	is.gd
9	Soo.gd	soo.gd
10	S2r.co	s2r.co

Apabila URL menggunakan salah satu dari bentuk singkatan dari shortener services di atas makan sistem akan memberikan nilai 1, sedangkan apabila URL tidak menggunakan salah bentuk singkatan tersebut maka sistem memberikan nilai -1.

Tabel 3.16. Nilai Fitur *Shortening Service*

Legitimate	*Phishing*
-1	1

d. Having At Symbol

Having At Symbol ini merupakan fitur yang berfungsi untuk mengecek apakah URL menggunakan simbol "@" atau tidak. Untuk melakukan ekstraksi pada fitur ini dilakukan dengan cara menggunakan fungsi find pada karakter dengan menggunakan python. Apabila tidak ditemukan simbol "@" pada URL maka sistem akan memberikan nilai -1, sedangkan apabila ditemukan simbol "@" pada URL akan diberikan nilai 1.

Tabel 3.17. Nilai Fitur *Having At Symbol*

Legitimate	*Phishing*
-1	1

e. Double Slash Redirecting

Keberadaan sebuah "//" pada sebuah URL berarti bahwa *user* akan dialihkan pada situs web lain seperti misalnya URL di bawah.

"http://www.legitimate.com//http://www.phishing.com"

Fitur ini mencoba mengecek lokasi dari "//", apabila URL berawal dengan "HTTP" maka otomatis posisi "//" akan berada posisi ke enam, sedangkan apabila URL berawal dengan "HTTPS" maka otomatis "//" akan berada pada posisi ke tujuh. Jadi apabila posisi "//" tidak berada pada posisi enam atau tujuh maka "//" berfungsi untuk mengalihkan *user* ke situs web lain.

Maka untuk melakukan ekstraksi fitur tersebut dilakukan dengan menggunakan fungsi *find*, apabila fungsi *find* menemukan "//" pada posisi lebih besar dari tujuh maka sistem akan memberikan nilai 1, sedangkan apabila posisi yang ditemukan lebih kecil sama dengan tujuh maka sistem akan memberikan nilai -1.

Tabel 3.18. Nilai Fitur *Double Slash Redirecting*

Legitimate	*Phishing*
-1	1

f. *Prefix Suffix*

Simbol tanda hubung (-) jarang digunakan dalam URL yang sah. *Phisher* cenderung menambahkan awalan dipisahkan oleh (-) ke nama domain sehingga pengguna merasa bahwa mereka berurusan dengan halaman web yang sah. Misalnya pada URL berikut ini:

"http://www.Confirme-paypal.com/."

Jadi pada fitur ini mencoba untuk melihat apakah hostname URL tersebut menggunakan tanda hubung (-) atau tidak. Untuk melakukan ekstraksi fitur tersebut dapat dilakukan dengan fungsi pada find pada hostname URL tersebut. Apabila ditemukan tanda hubung (-) maka sistem akan memberikan nilai 1, sedangkan apabila tidak ditemukan tanda hubung (-) maka sistem memberikan nilai -1.

Tabel 3.19. Nilai Fitur *Prefix Suffix*

Legitimate	*Phishing*
-1	1

g. *Having Sub Domain*

Mari asumsikan bahwa ada URL seperti berikut: "http://www.hud.ac.uk/students/". Nama domain tersebut menyertakan country-code top-level domains (ccTLD), yang dalam URL tersebut adalah "uk". Bagian "ac" adalah singkatan untuk "akademis", gabungan "ac.uk" disebut domain tingkat

kedua atau second-level domain (SLD) dan "hud" adalah nama sebenarnya dari domain tersebut. Untuk menghasilkan aturan untuk mengekstraksi fitur ini, pertama-tama kita harus menghilangkan (www.) dari URL yang sebenarnya merupakan sub domain itu sendiri. Kemudian, kita harus menghapus (ccTLD) jika ada. Maka selanjutnya dapat dihitung titik-titik yang tersisa. Jika jumlah titik lebih besar dari satu, maka URL diklasifikasikan sebagai "*Suspicious*" karena memiliki satu sub domain. Namun, jika titik-titik lebih besar dari dua, itu diklasifikasikan sebagai "*Phishing*" karena akan memiliki beberapa sub domain. Sedangkan apabila jumlah titik yang ditemukan hanya satu maka URL dapat dikatan "*Legitimate*". Untuk melakukan ekstraksi fitur tersebut menggunakan pemrograman python pada sistem dilakukan dengan pertama-tama mengecek (ccTLD) menggunakan daftar (ccTLD) pada tabel berikut dengan menggunakan fungsi *find*:

Tabel 3.20. Daftar country-code Top-Level Domain (ccTLD)

NO.	Country-code Top-Level Domain (ccTLD)	Bentuk Domain
1	Lingkungan akademik/perguruan tinggi	.ac.
2	Kepentingan komersial	.co.
3	Situs pemerintahan desa	.desa.
4	Organisasi	.or.
5	Jasa telekomunikasi	.net.
6	Perseorangan maupun organisasi	.web.
7	Sekolah	.sch.
8	Pemerintah	.go.

Setelah dilakukan pengecekan dan menghasilan tidak ditemukan salah satu dari (ccTLD) di atas maka langkah selanjutnya yaitu menghitung jumlah titik pada domain. Apabila jumlah titik sama dengan satu maka sistem akan memberikan nilai -1, sedangkan apabila jumlah titik sama dengan dua maka sistem akan memberikan nilai 0, dan apabila jumlah titik lebih dari dua sistem akan memberikan nilai 1. Tapi apabila pada saat melakukan pengecekan (ccTLD) ditemukan salah satu dari bentuk (ccTLD) di atas maka aturan yang dipakai adalah aturan pertama yaitu tidak ditemukan (ccTLD) dikurangi jumlah titik satu.

Tabel 3.21. Nilai Fitur *Having Sub Domain*

Legitimate	*Suspicious*	*Phishing*
-1	0	1

h. Domain Registration Length

Berdasarkan kenyataan bahwa *website phishing* hanya memiliki masa aktif yang singkat, sedangkan *website* yang terpercaya memiliki masa aktif yang lebih lama. Dalam dataset dari Kaggle Dataset ditemukan bahwa domain penipuan memiliki masa aktif terpanjang yaitu hanya satu tahun saja. Maka pada fitur ini memcoba untuk mengecek masa aktif sebuah domain, apabila domain tersebut hanya memiliki masa aktif lebih kecil sama dengan satu tahun maka dikategorikan "*Phishing*", sedangkan

apabila lebih dari satu tahun dikategorikan sebagai bukan *phishing*.

Pada sistem ini dilakukan ekstraksi fitur dengan cara menggunakan data *whois* yaitu *updated_date* dan *expiration_date*. Untuk mendapatkan tahun masa aktif *website* tersebut dilakukan dengan *expiration_date* dikurangi dengan *updated_date*. Namun apabila sebuah *website* tidak memiliki data *whois updated_date* maka digunakan data *creation_date* sebagai pengganti *updated_date*. Apabila hasil pengurangan antara *expiration_date* dikurangi *updated_date* atau *creation_date* lebih kecil sama dengan satu tahun maka sistem akan memberikan nilai 1, sedangkan apabila lebih dari satu tahun maka akan diberikan nilai -1. Tapi apabila tidak ditemukan sama sekali data *whois* atau tidak adanya data *creation_date*, *updated_date*, *expiration_date* maka sistem akan langsung memberikan nilai 1.

Tabel 3.22. Nilai Fitur *Domain Registration Length*

Legitimate	*Phishing*
-1	1

i. HTTPS Token

Fitur ini mencoba untuk mendeteksi *phishing* dengan mengecek adanya penggunaan “HTTPS” pada bagian URL seperti pada bagian domain untuk mengecoh *user* agar merasa bahwa *website*

yang dikunjungi tersebut merupakan *website* yang aman dan terpercaya. Misalnya seperti URL berikut:

"http://https-www-paypal-it-webapps-mpp-home.soft-hair.com/"

Apabila ditemukan penggunaan "HTTPS" tersebut pada URL maka sistem akan memberikan nilai 1 sedangkan apabila tidak ditemukan maka sistem akan memberikan nilai -1.

Tabel 3.23. Nilai Fitur *HTTPS Token*

Legitimate	*Phishing*
-1	1

j. Submitting Information to Email

Web yang berbentuk formulir memungkinkan pengguna untuk mengirimkan informasi pribadinya yang diarahkan ke server untuk diproses. Dari sini *phisher* memungkinkan untuk mengarahkan informasi *user* ke email pribadinya. Untuk melakukan hal tersebut *phisher* biasa menggunakan fungsi "mail()" di PHP dan fungsi "mailto:".

Untuk mendeteksi penggunaan kedua fungsi tersebut maka sistem mencoba untuk mengakses *source page* dari sebuah *website* dan kemudian mencari adanya penggunaan fungsi "mail()" dan "mailto:". Apabila sistem menemukan penggunaan salah satu fungsi tersebut maka sistem akan memberikan nilai 1 dan jika tidak ditemukan akan diberikan nilai -1.

Tabel 3.24. Nilai Fitur *Submitting Information to Email*

Legitimate	*Phishing*
-1	1

k. Right Click

Phisher biasa menggunakan JavaScript untuk mematikan fungsi *RightClick*, sehingga *user* tidak dapat melihat dan menyimpan source page dari sebuah *website*. Dalam sistem ini mencoba untuk mengekstrak fitur ini dengan cara mengakses source web tersebut kemudian mencari adanya penggunaan "contextmenu" pada source page tersebut. Apabila ditemukan penggunaan "contextmenu" maka sistem akan memberikan nilai 1 dan jika tidak ditemukan akan diberikan nilai -1.

Tabel 3.25. Nilai Fitur *Right Click*

Legitimate	*Phishing*
-1	1

l. IFrame

Iframe merupakan *tag* HTML yang digunakan untuk menampilkan *website* tambahan sehingga dapat tampil pada web tersebut. Untuk mengakses fitur tersebut, sistem terlebih dahulu mengakses *source page* dari *website* tersebut kemudian mencari adanya penggunaan tag "iframe" pada *website* tersebut. Apabila ditemukan penggunaan tag tersebut maka sistem akan

memberikan nilai 1 dan jika tidak ditemukan maka akan diberikan nilai -1.

Tabel 3.26. Nilai Fitur *IFrame*

Legitimate	*Phishing*
-1	1

m. Age of Domain

Di dalam fitur sistem akan mencoba untuk melihat usia dari domain yang digunakan oleh sebuah *website*. Untuk mengekstrak data tersebut dapat dilakukan melalui pengecekan pada WHOIS. Apabila dalam pengecekan WHOIS ditemukan usia *website* lebih besar atau sama dengan enam bulan maka sistem akan memberikan nilai -1, sedangkan di bawah nilai tersebut akan diberikan nilai 1. Sistem melakukan pengecekan ini dengan cara mengurangi waktu sekarang dengan waktu pembuatan *website* tersebut.

Tabel 3.27. Nilai Fitur *Age of Domain*

Legitimate	*Phishing*
-1	1

n. DNS Record

Fitur ini berfungsi untuk mengecek apakah *website* tersebut memiliki *DNS record* atau tidak. Di dalam sistem ini ekstraksi fitur ini dilakukan dengan pengecekan pada beberapa *DNS record* yaitu sebagai berikut:

Tabel 3.28. *DNS Record*

No.	DNS Record
1	CNAME
2	MX
3	NS
4	PTR
5	SRV
6	SOA
7	TXT

Setelah melakukan pengecekan dengan *DNS record* di atas maka akan dicek apakah *website* yang diprediksi memiliki *DNS record* pada salah satu pengecekan di atas. Apabila ditemukan memilik *DNS record* pada salah satu pengecekan di atas makan akan diberikan nilai 1, sedangkan apabila tidak memiliki sama sekali akan diberikan nilai -1.

Tabel 3.29. Nilai Fitur *DNS Record*

Legitimate	*Phishing*
1	-1

o. Website Traffic

Fitur ini mengukur popularitas situs web dengan menentukan jumlah pengunjung dan jumlah halaman yang mereka kunjungi. Data fitur ini didasarkan pada database ALEXA, sehingga untuk mengecek *website traffic* tersebut sistem akan menggunakan database ALEXA sebagai acuan. Jika sistem menemukan peringkat website di bawah 100.000 maka sistem akan memberika nilai 0, sedangkan jika peringkatnya ditemukan lebih

besar dari 100.000 maka sistem akan memberikan nilai -1, dan jika data website tidak ditemukan dalam database ALEXA akan diberikan nilai 1.

Tabel 3.30. Nilai Fitur *Website Traffic*

Legitimate	*Suspicious*	*Phishing*
0	-1	1

p. Statistical Report

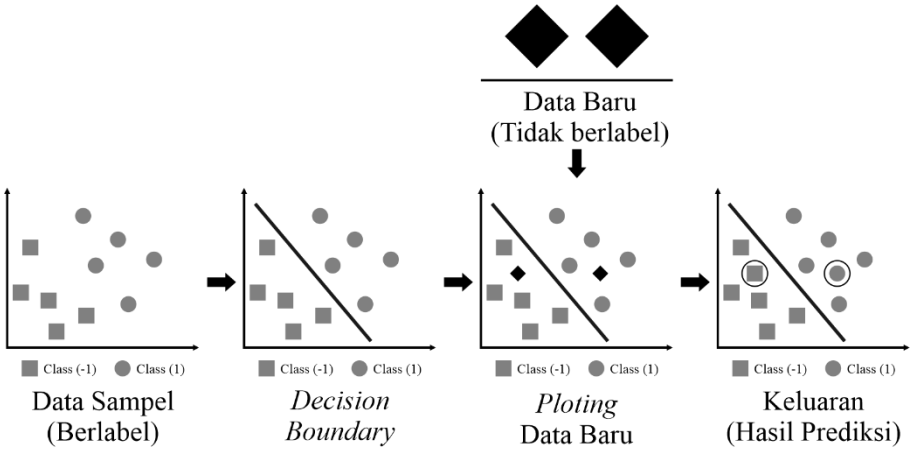
Fitur statistical *report* ini merupakan fitur yang didasarkan pada "Top 10 Domains" dan "Top 10 IPs" pada *website PhishTank*, serta "Top 50" *IP address* pada *StopBadware*. Jadi untuk mengekstrak fitur tersebut, pertama-tama sistem akan mengambil data-data di atas pada *PhishTank* dan *StopBadware* kemudian mencocokkannya dengan *website* yang akan diprediksi. Apabila *website* tersebut ditemukan di dalam data PhishTank ataupun StopBadware maka akan diberikan nilai 1, sedangkan jika tidak ditemukan akan diberikan nilai -1.

Tabel 3.31. Nilai Fitur *Statistical Report*

Legitimate	*Phishing*
-1	1

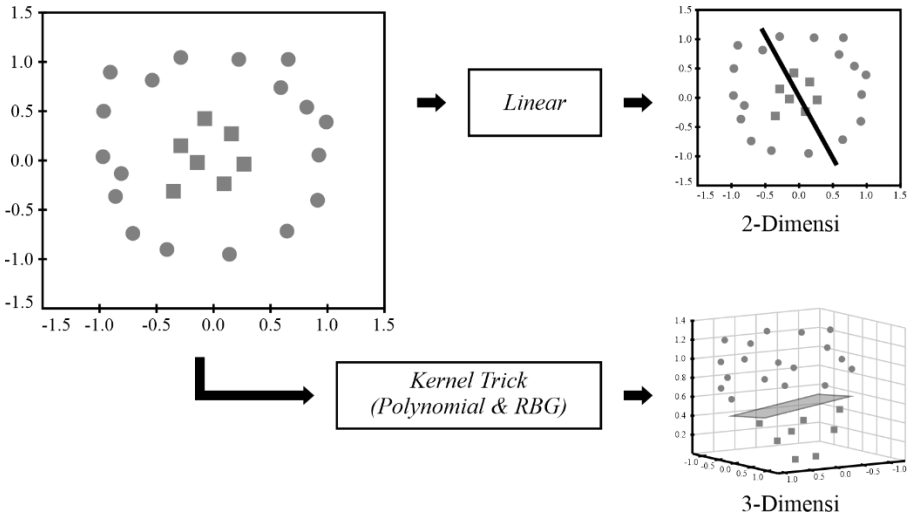
4. Model *Support Vector Machine* (SVM)

Seperti yang dijelaskan pada bab sebelumnya, SVM sebenarnya merupakan algoritma klasifikasi yang bersifat *linear* dengan menggunakan metode mencari *hyperplane* terbaik dalam memisahkan setiap *class*. Secara sederhana proses klasifikasi SVM dapat dilihat pada Gambar 3.9.



Gambar 3.9. Proses prediksi SVM

Seperti yang telah dijelaskan sebelumnya bahwa SVM sebenarnya merupakan algoritma klasifikasi yang bersifat *linear*. Sehingga apabila mendapatkan data yang *non-linear* akan terjadi kesulitan dalam menentukan *hyperplane* terbaik. Seperti contoh data pada gambar 3.6. yang apabila digunakan klasifikasi *linear* akan menghasilkan *hyperplane* yang tidak ortimal. Oleh karena itu, dibuatlah sistem *kernel trick*. Yang dari *kernel trick* ini lahirlah *kernel polynomial* dan *kernel Gaussian radial basis function* (RBF) yang dapat memisahkan *class* dalam bentuk data yang berdimensi tinggi. Untuk lebih memahami peran *kernel trick* dapat dilihat pada gambar 3.6.



Gambar 3.10. Perbandingan *kernel trick* dan *linear*

5. *Decision Function*

Decision function pada sistem ini merupakan sebuah fungsi yang bertujuan untuk menentukan hasil klasifikasi pada sistem. Karena *class* pada dataset yaitu -1 (*legitimate*) dan 1 (*phishing*) maka otomatis *threshold* sistem akan berada pada nilai 0. Jadi apabila hasil perhitungan *decision function* > 0 maka data akan diklasifikasikan sebagai *phishing* dan sebaliknya apabila *decision function* < 0 data akan diklasifikasikan sebagai *legitimate.* Hasil klasifikasi disini memiliki dua keluaran yaitu:

a. Persentase Tingkat *Phishing*

Persentase tingkat *phishing* ini merupakan keluaran yang berupa persentase yang menyatakan tingkat *phishing* dari sebuah *website*. Persentase tingkat phishing dihitung berdasarkan hasil dari perhitungan *decision function.* Semakin mendekati nilai -1 hasil perhitungan *decision function* maka semakin rendah tingkat

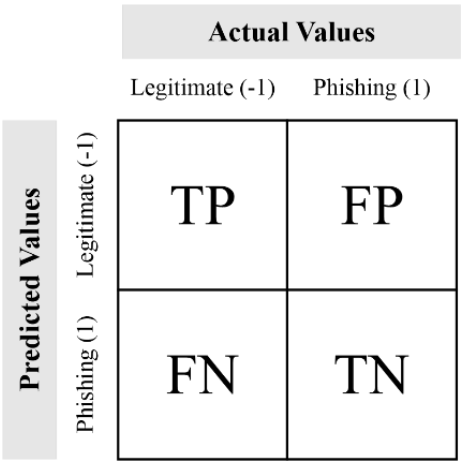
phishing yang dimiliki oleh data tersebut, begitupun sebaliknya. Semakin mendekati nilai 1 hasil perhitungan *decision function* maka semakin tingggi tingkat phishing data tersebut. Hasil perhitungan persentase *phishing* pada sistem ini dikeluarkan dalam bentuk persen (%). Dengan bentuk keluaran seperti ini dimaksudkan agar *user* dapat memiliki pertimbangan, apakah *website* tersebut *phishing* atau bukan.

b. Prediksi

Keluaran prediksi ini merupakan hasil klassifikasi dari algoritma yang digunakan, keluarannya berupa -1 (legitimate) dan 1 (*phishing*). Keluaran ini berupa hasil prediksi dari sistem apakah *website* tersebut merupakan *phishing* atau bukan. Prediksi ini didapatkan dari nilai yang dikeluarkan oleh *decision function*, apabila *decision fuction* > 0 maka data akan secara otomatis diklasifikasikan sebagai *phishing.* Dan sebaliknya apabila nilai *decision function* < 0, data akan diklasifikasikan sebagai *legitimate.*

3.6. Analisis Kerja Sistem

Analisis kerja sistem prediksi *website phishing* dilakukan dengan menghitung nilai akurasi berdasarkan ketepatan prediksi yang dilakukan oleh sistem. Perhitungan nilai akurasi dilakukan menggunakan bantuan *confusion matrix* seperti gambar 3.11. berikut.



Gambar 3.11. *Confusion matrix*

Confusion matrix terdiri atas empat bagian, yaitu:

1. True Positif : yaitu jumlah prediksi benar dari data positif.
2. False Positif : yaitu jumlah prediksi salah dari data positif.
3. False Negatif : yaitu jumlah prediksi salah dari data negatif.
4. True Negatif : yaitu jumlah prediksi benar dari data negatif.

Untuk menghitung nilai akurasi sistem, dapat dilakukan dengan persamaan (3.17) berikut.

$$Akurasi = \left(\frac{TP + TN}{TP + FP + FN + TN}\right) \times 100\% \qquad (3.17)$$

BAB IV

HASIL DAN PEMBAHASAN

4.1. Hasil Penelitian

4.1.1. Evaluasi

Pada bagian ini, disajikan hasil kinerja sistem deteksi *phishing* dengan menggunakan algoritma *Support Vector Machine* (SVM). Pada penelitian ini, *confusion matrix* digunakan untuk membantu dalam melakukan perhitungan akurasi sistem. Tabel *confusion matrix* untuk masing-masing percobaan dengan metode optimasi parameter dapat dilihat pada lampiran. Adapun jumlah data yang digunakan sebanyak 11055 data, yang dibagi dengan metode *10-fold cross-validation*.

Pada penelitian ini, dilakukan percobaan terhadap tiga algoritma dengan optimasi parameter pada setiap algoritma. Ketiga algoritma tersebut adalah:

1. Support Vector Machine (SVM)

a. Kernel Linear

1) Nilai parameter *C* sebesar **1.0**, **1.5**, **2.5**, dan **5.0**.

b. Kernel Polynomial

1) Nilai parameter *C* sebesar **1.0**, **1.5**, **2.5**, dan **5.0**.

2) Nilai parameter *degree* sebesar **1**, **3**, **5**, **7**, dan **9**.

c. Kernel RBF

1) Nilai parameter *C* sebesar **1.0**, **1.5**, **2.5**, dan **5.0**.

2) Nilai parameter *gamma* sebesar **0.1**, **0.5**, **1.0**, **1.5**, dan **2.0**.

2. Decision Tree

a. Nilai parameter *criterion* adalah *gini* dan *entropy*.

b. Nilai parameter *max_depth* sebesar **1**, **5**, **10**, **15**, dan **20**.

3. K-Nearest Neighbors

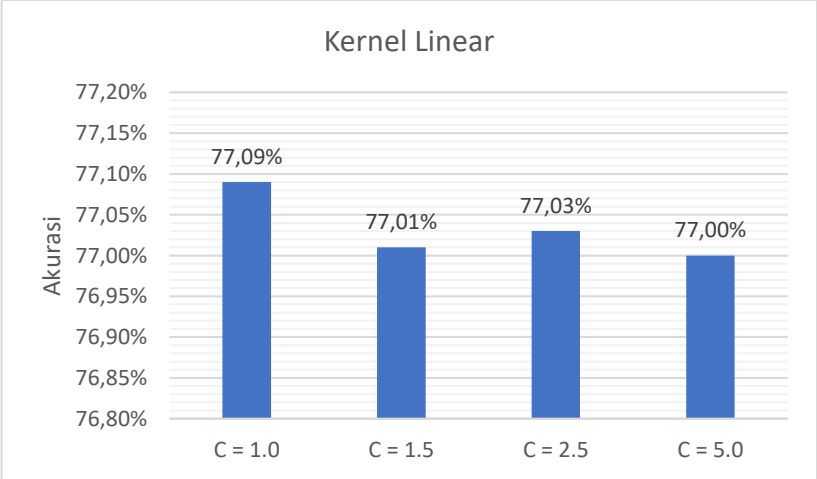
a. Nilai parameter *weights* adalah *uniform* dan *distance*.

b. Nilai parameter *n_neighbors* sebesar **1**, **5**, **10**, **15**, dan **20**.

Adapun hasil kinerja sistem untuk masing-masing percobaan ditunjukkan sebagai berikut.

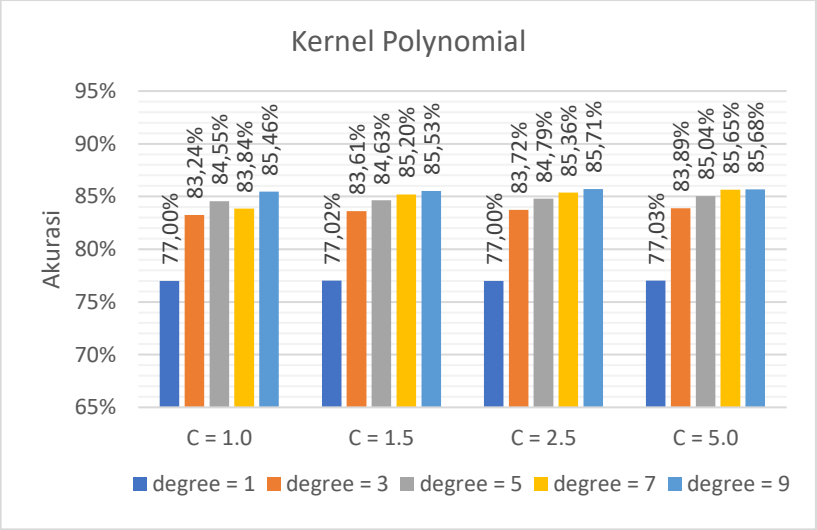
1. *Support Vector Machine* (SVM)

a. *Kernel Linear*



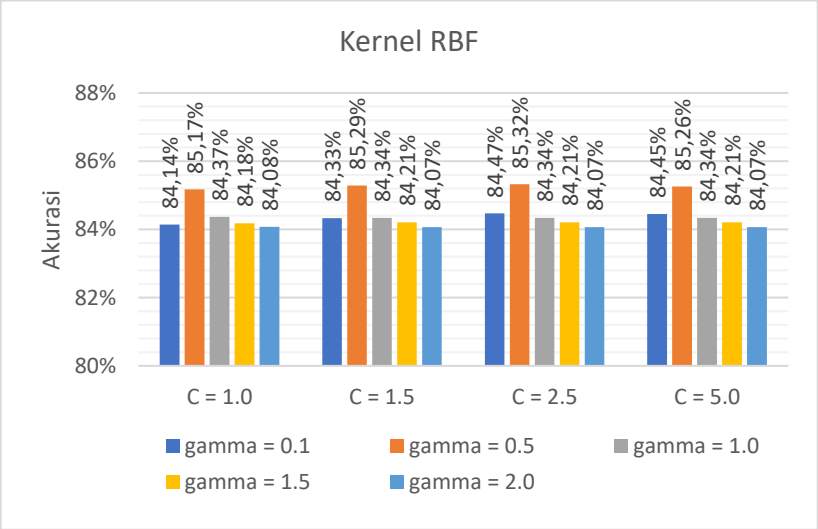
Gambar 4.1. Grafik Perbandingan Akurasi berdasarkan Nilai *C*

b. *Kernel Polynomial*



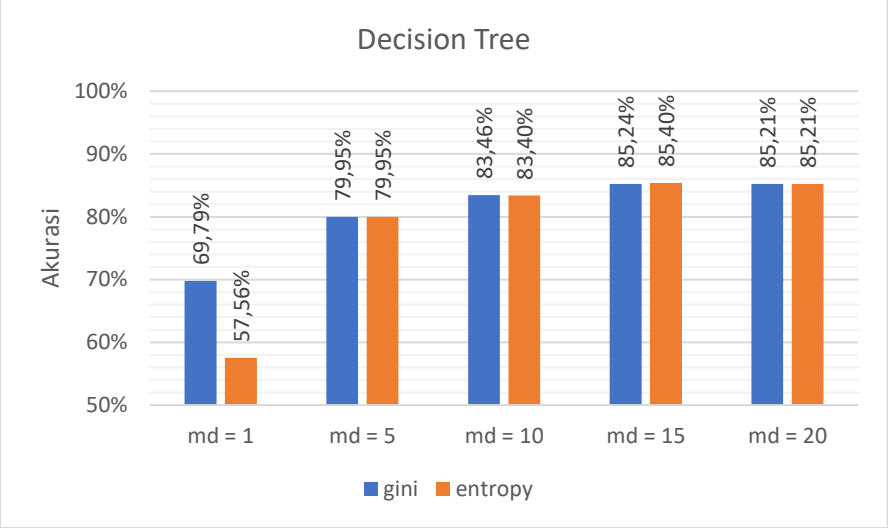
Gambar 4.2. Grafik Perbandingan Akurasi berdasarkan Nilai *degree* dan *C*

c. *Kernel RBF*



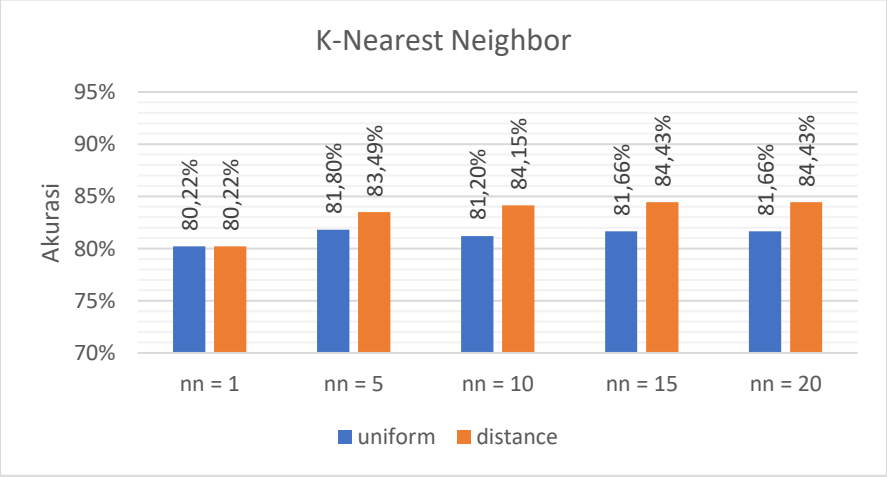
Gambar 4.3. Grafik Perbandingan Akurasi berdasarkan Nilai *gamma* dan *C*

2. *Decision Tree*



Gambar 4.4. Grafik Perbandingan Akurasi berdasarkan Nilai *max_depth* dan *criterion*

3. *K-Nearest Neighbors*



Gambar 4.5. Grafik Perbandingan Akurasi berdasarkan Nilai *n_neighbors* dan *weights*

Berdasarkan tabel yang telah dipaparkan, dapat dilihat bahwa akurasi baik sebesar **85.71%** diperoleh pada algoritma SVM *kernel polynomial* dengan nilai *degree* 9 serta nilai *C* 2.5.

4.1.2. Implementasi

Pada bagian ini, disajikan hasil pengimplementasian sistem deteksi *website phishing* dengan menggunakan algoritma *support vector machine* (SVM). Pada pengimplementasiannya, sistem telah dapat menerima masukkan dari *user* berupa URL yang akan dideteksi. Setelah URL dimasukkan oleh *user* maka sistem akan melakukan proses ekstraksi fitur terhadapt URL tersebut. Hasil ekstraksi fitur tersebut akan disimpan dalam bentuk *comma separated value* (CSV). Contoh hasil ekstraksi fitur dari sistem dapat dilihat pada Tabel 4.1.

Tabel 4.1. Hasil Ekstraksi Fitur

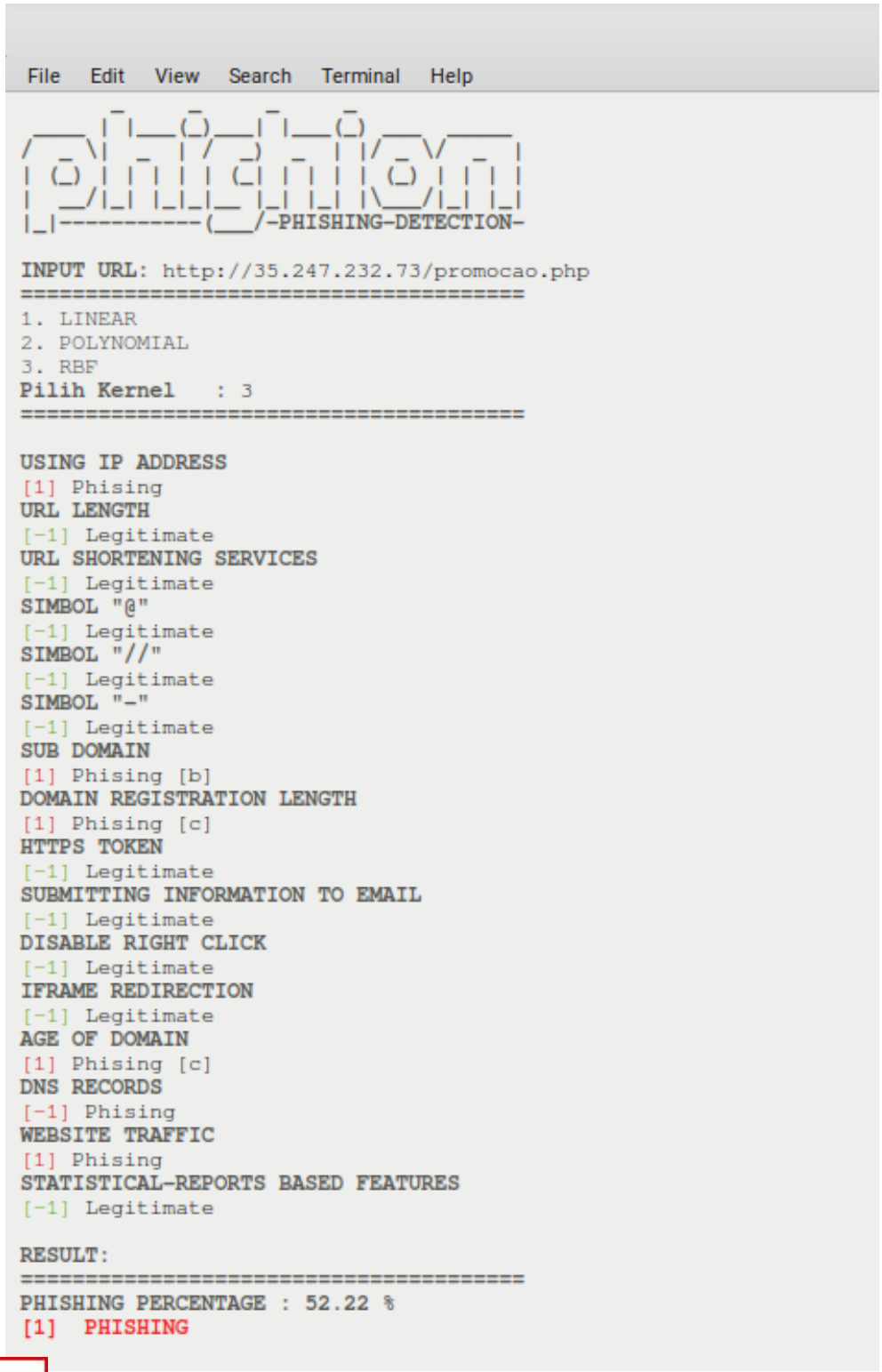
Having IP Address	URL Length	Shortining Service	Having At Symbol	Double Slash Redirecting	Prefix Suffix	Having Sub Domain	Domain Registration Length	HTTPS Token	Submitting to Email	Right Click	Iframe	Age of Domain	DNS Record	Web Traffic	Statistical Report
-1	1	1	1	-1	-1	-1	-1	-1	-1	1	1	-1	-1	-1	-1

Dari hasil ekstraksi fitur tersebut akan dilakukan klasifikasi dengan menggunakan model SVM yang menggunakan data latih dari Kaggle dataset. Dan dengan model tersebut maka sistem akan dapat mengklasifikasikan URL yang dimasukkan sebelumnya. Berikut ini beberapa URL yang telah dilakukan pengujian dengan menggunakan sistem pendeteksi *phishing* yang

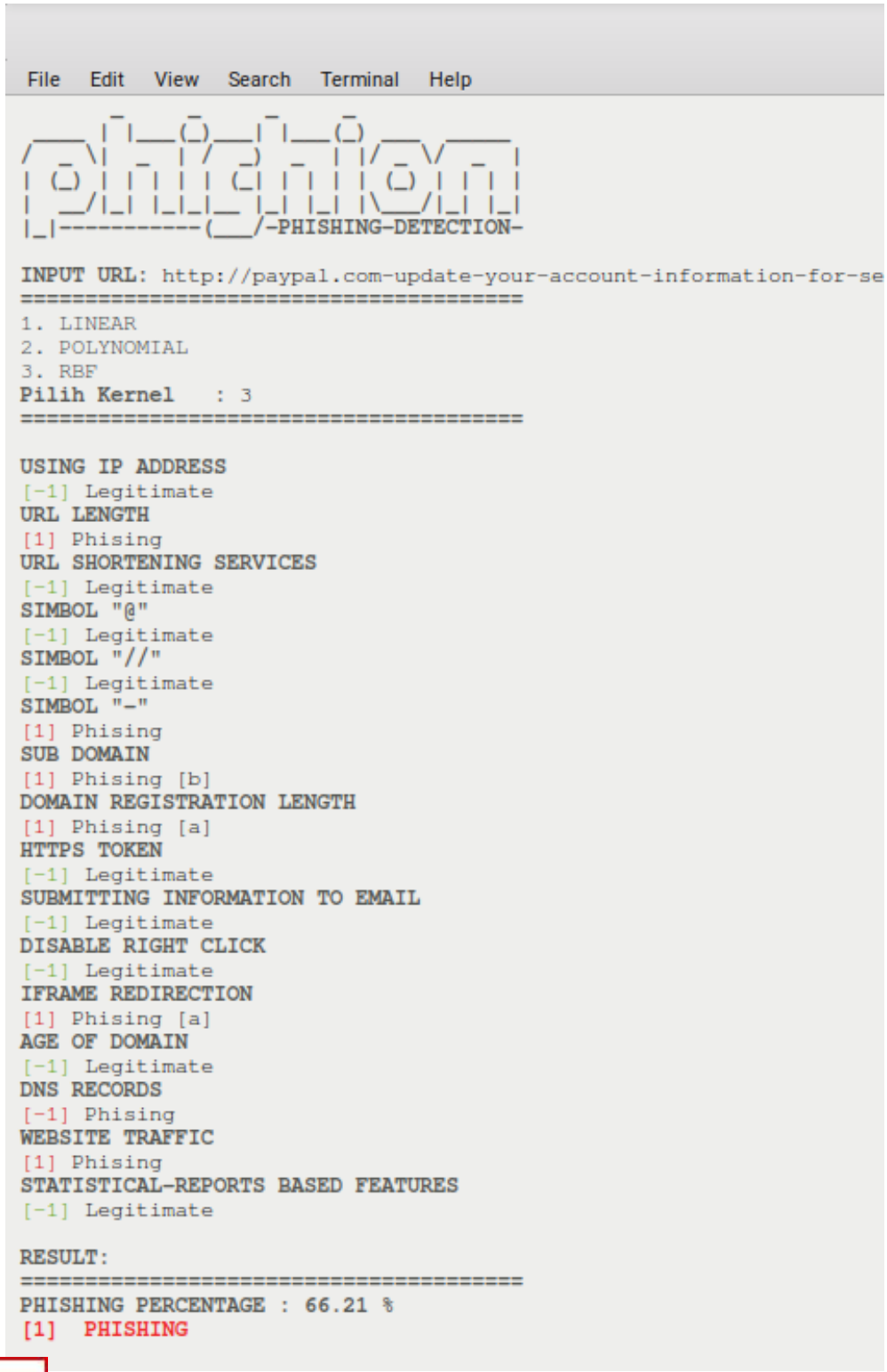
h dibuat.

mbar 4.6. Hasil Implementasi Program Pendeteksi *Website Phishing* pada website *phishing*

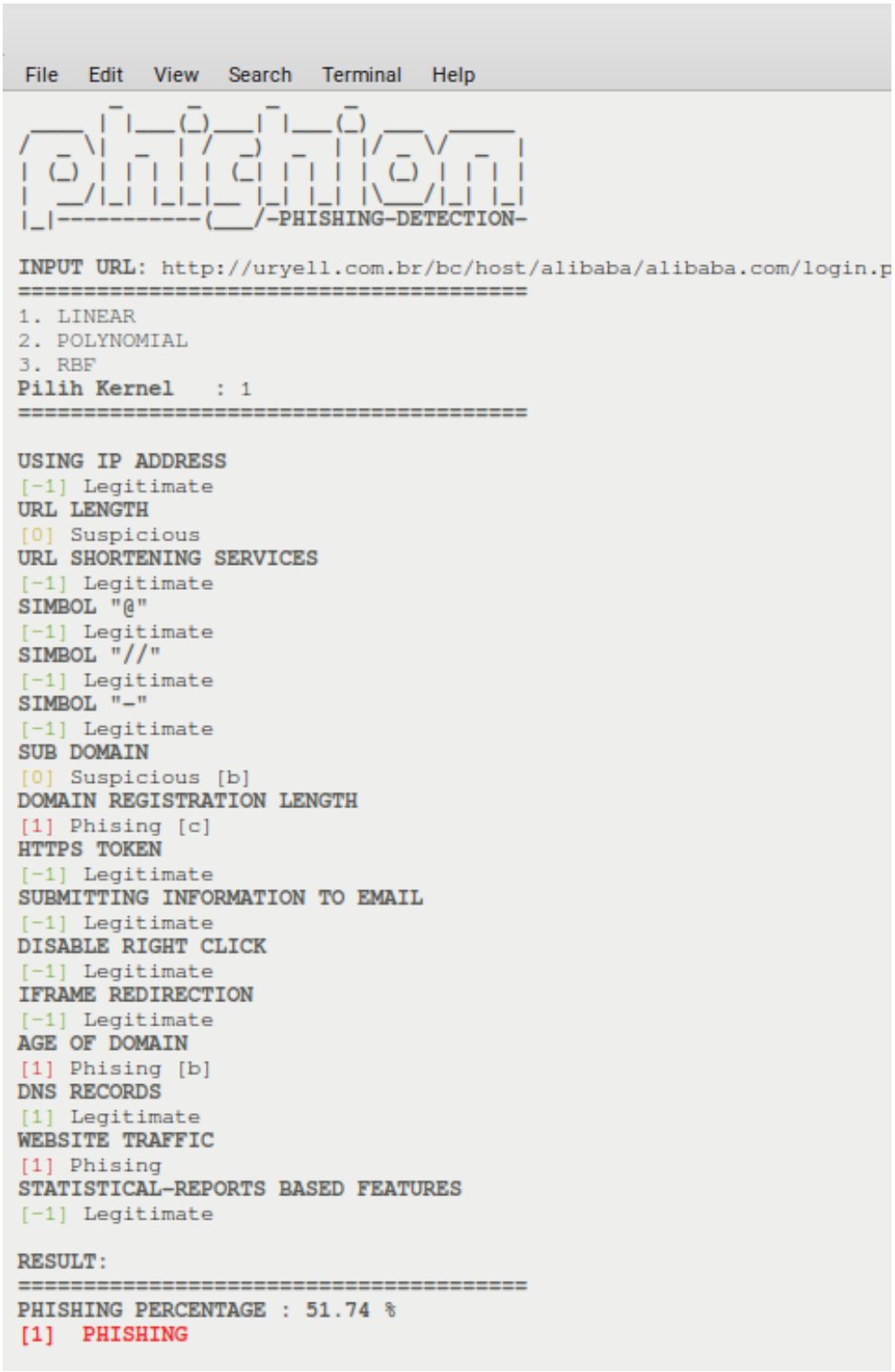
mbar 4.7. Hasil Implementasi Program Pendeteksi *Website Phishing* pada website *phishing*



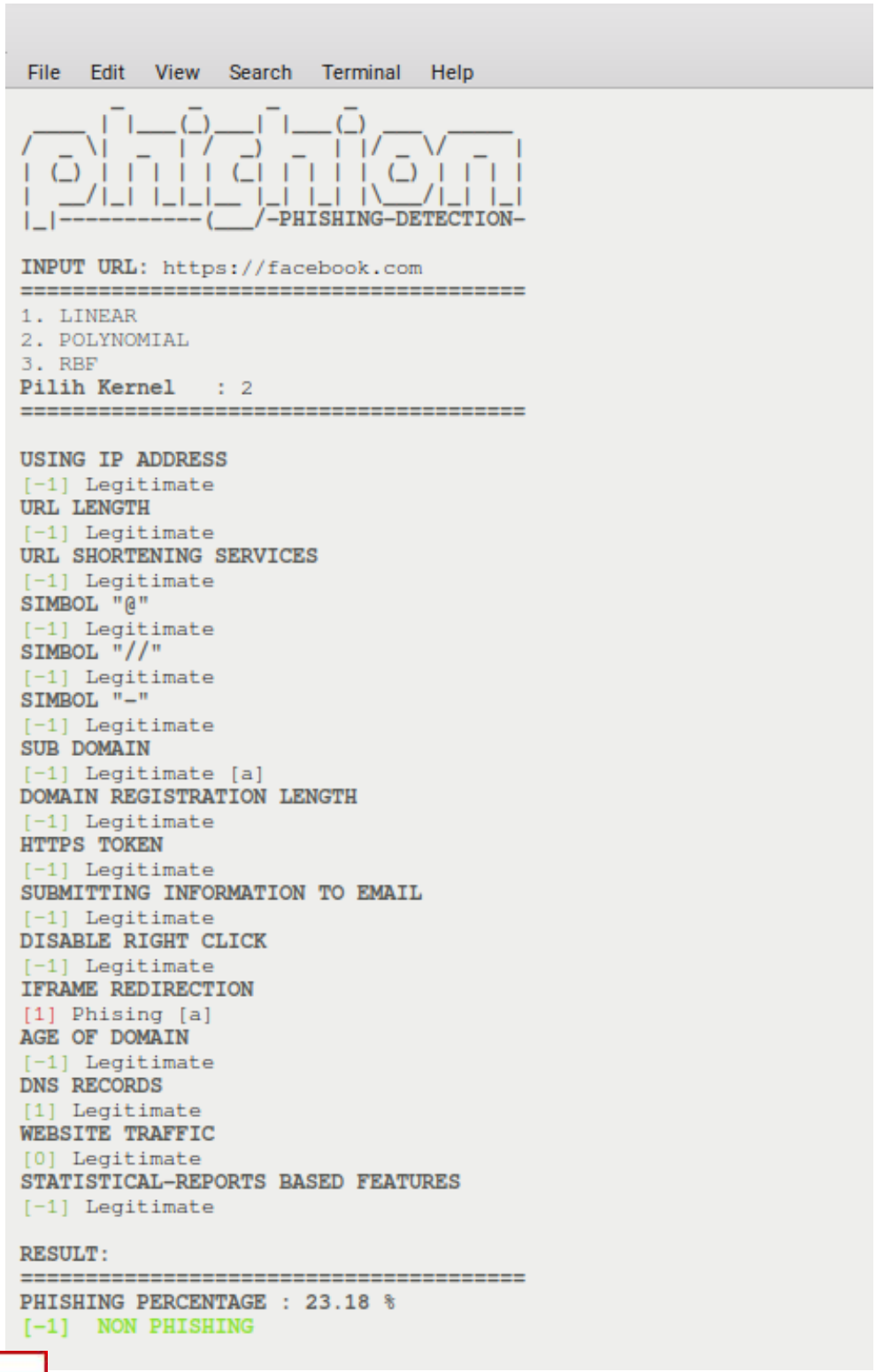
mbar 4.8. Hasil Implementasi Program Pendeteksi *Website Phishing* pada website *phishing*



mbar 4.9. Hasil Implementasi Program Pendeteksi *Website Phishing* pada website *phishing*

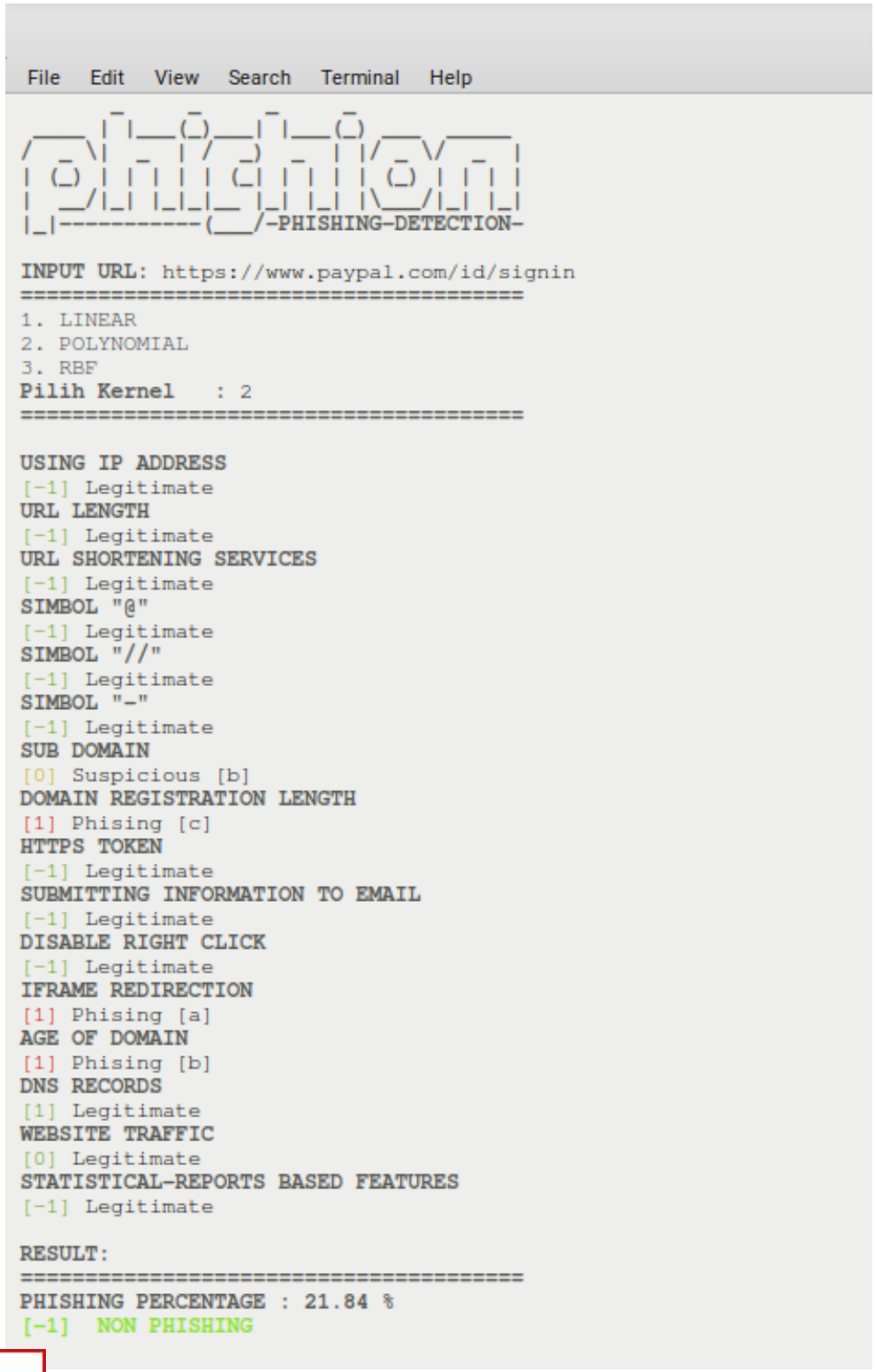


Gambar 4.10. Hasil Implementasi Program Pendeteksi *Website Phishing* pada website *phishing*



Gambar 4.11. Hasil Implementasi Program Pendeteksi *Website Phishing* pada website *non phishing*

Gambar 4.12. Hasil Implementasi Program Pendeteksi *Website Phishing* pada website *non phishing*



Gambar 4.13. Hasil Implementasi Program Pendeteksi *Website Phishing* pada website *non phishing*

Gambar 4.14. Hasil Implementasi Program Pendeteksi *Website Phishing* pada website *non phishing*

Gambar 4.15. Hasil Implementasi Program Pendeteksi *Website Phishing* pada website *non phishing*

4.2. Pembahasan

4.2.1. Evaluasi

1. Support Vector Machine (SVM)

a. Pengaruh nilai C terhadap hasil kinerja algoritma

Parameter *C* merupakan besaran *penalty* yang diberikan terhadap *error* klasifikasi. Pada *kernel linear* didapatkan akurasi terbesar pada nilai *C* 1.0. Dengan melihat data pada Gambar 4.1. dapat dikatakan bahwa untuk *kernel linear* dengan dataset pada penelitian ini kurang baik menggunakan nilai *C* yang besar. Hampir sama dengan *kernel linear*, hasil akurasi tertinggi yang didapatkan dari *kernel polynomial* dan *RBF* adalah pada nilai *C* 2.5 dengan nilai akurasi masing-masing 85.71% dan 85.32%.

Dari hasil percobaan dengan optimasi parameter *C* pada ketiga *kernel* di atas dapat disimpulkan bahwa untuk dataset *website phishing* yang digunakan pada penelitian kurang baik menggunakan nilai *C* yang terlalu besar.

b. Pengaruh nilai *degree* terhadap hasil kinerja algoritma

Parameter *degree* merupakan parameter yang hanya dapat dioptimasi pada *kernel polynomial*. Dari hasil pengujian, akurasi terbaik diperoleh pada nilai *degree* 9 dengan nilai *C* 2.5 sekaligus menjadi akurasi terbaik diantara algoritma

yang lainnya. Dan dengan meperhatikan akurasi di setiap nilai *degree*, terlihat terjadi peningkatan akurasi yang cukup signifikan. Yang dengan nilai *degree* terendah diperoleh 77.00% dapat meningkat menjadi 85.00%. Sehingga dari hasil tersebut dapat disimpulkan bahwa dengan mengoptimalkan parameter *degree* pada nilai tertentu dapat meningkatkan akurasi sistem.

c. Pengaruh nilai *gamma* terhadap hasil kinerja algoritma

Parameter *gamma* merupakan optimasi parameter pada algoritma SVM *kernel* RBF dengan menggunakan nilai, 0.1, 0.5, 1.0, 1,5, 2.0. Dari hasil pengujian diperolah akurasi terbaik pada nilai *gamma* 0.5 dengan nilai *C* 2.5. Dengan memperhatikan akurasi disetiap nilai *gamma* dan *C*, dapat ditarik kesimpulan bahwa parameter *gamma* optimal pada nilai 0.5.

2. Decision Tree

Pada percobaan dengan menggunakan *decision tree* digunakan dua *criterion* dalam melakukan *splitting* data yaitu *gini* dan *entropy*. Serta nilai *max_depth* sebesar 1, 5, 10, 15, dan 20.

Berdasarkan Gambar 4.4. dengan menggunakan *gini* didapatkan akurasi tertinggi pada nilai *max_depth* 15 yaitu

85.24%. sedangkan pada *entropy* akurasi tertinggi didapatkan pada nilai *max_depth* 15 yaitu 85.40%. Dari hasil ini dapat disimpulkan bahwa untuk dataset pada penelitian ini lebih cocok digunakan *criterion entropy*. Dan tidak selalu dengan *max_depth* yang besar akan didapatkan akurasi yang lebih baik tergantung dengan jumlah fitur yang digunakan.

3. K-Nearest Neighbors

Pada percobaan yang dilakukan terhadap algoritma KNN digunakan nilai *weights* yaitu *unform* dan *distance*. Sedangkan *n_neighbors* sebesar 1, 5, 10, 15, dan 20. Berdasarkan pada Gambar 4.5., dengan menggunakan nilai *uniform* didapatkan akurasi tertinggi pada nilai *n_neighbors* 5, sedangkan dengan menggunakan nilai distance didapatkan akurasi tertinggi pada nilai *n_neighbors* 20. Hasil ini menunjukkan bahwa dengan *uniform* tidak selamanya semakin besar nilai *n_neighbors* maka semakin besar pula akurasi yang didapatkan. Berbeda halnya dengan *distance* semakin besar nilai *n_neighbors* semakin baik pula akurasi yang didapatkan.

Pada *uniform*, bobot yang diberikan pada setiap *n_neighbors* adalah sama. Jadi, setiap *n_neighbors* memiliki pengaruh yang sama terhadap titik. Dengan setiap *n_neighbors* memiliki pengaruh yang sama maka apabila *n_neighbors* yang dipilih terlalu besar bisa menyebabkan klasifikasi menjadi kurang

baik, hal ini dikarenakan banyaknya *n_neighbors* yang berpengaruh terhadap titik. Berbeda dengan *distance* semakin besar *n_neighbors* semakin baik akurasinya. Hal ini dikarenakan bobot yang diberikan terhadap setiap *n_neighbors* tidak sama melainkan ditentukan berdasarkan jaraknya dari titik. Jadi meskipun nilai *n_neighbors* besar, yang akan sangat berpengaruh terhadap titik hanyalah yang paling dekat dengan titik.

4.2.2. Implementasi

Pada sistem implementasi ada beberapa hal yang membedakannya dengan sistem evaluasi yaitu pada sistem implementasi telah ada kontak dengan *user* yang akan memasukkan URL untuk dilakukan deteksi *website phishing.* Dan di dalam sistem evaluasi terdapat proses yang berperan penting yaitu ekstraksi fitur. Dengan adanya ektraksi fitur, sistem akan dapat mengekstrak fitur-fitur yang dibutuhkan sesuai dengan dataset yang digunakan dari *Kaggle dataset*. Hasil ekstraksi fitur inilah yang nantinya akan digunakan sebagai data uji sekaligus sebagai data yang akan diklasifikasikan oleh sistem dengan menggunakan algoritma *macine learning support vector machine* (SVM).

BAB V

PENUTUP

5.1. Kesimpulan

Dari hasil analisis yang telah dilakukan dalam pengujian sistem deteksi *website phishing* dengan menggunakan metode *machine learning*, dapat disimpulkan bahwa:

1. Sistem deteksi *website phishing* dibuat dengan menggunakan algoritma *machine learning Support Machine Learning* (SVM) sebagai algoritma untuk mendeteksi website phishing. Sebelum URL yang menjadi masukan sistem diproses dengan menggunakan algoritma *Support Vector Machine* (SVM), terlebih dahulu akan dilakukan proses ekstraksi fitur pada URL yang telah dimasukkan. Metode yang digunakan dalam melakukan ekstraksi fitur berbeda-beda untuk setiap fitur yang akan diekstrak. Metode yang digunakan antara lain melakukan ekstraksi URL *Obfuscation*, ekstraksi data Whois, ekstraksi *DNS Record*, ekstraksi data dari Alexa database, serta ekstraksi data pada *website* PhishTank dan StopBadware.
2. Hasil pengujian dengan menggunakan *10-fild cross-validation* dengan algoritma *machine learning Support Vector Machine* (SVM), *Decision Tree*, dan *K-Nearest Neighbor* serta proses optimasi parameter pada setiap algoritma didapatkan akurasi terbaik dari masing-masing algoritma yakni SVM *kernel linear* 77.09% dengan nilai parameter *C* 1.0, *kernel polynomial* 85.71% dengan nilai parameter *degree* 9 dan *C*

2.5, *kernel* RBF 85,32% dengan nilai parameter *gamma* 0.5 dan *C* 2.5, *decision tree* 85.40% dengan nilai parameter *criterion entropy* dan *max_depth* 15, dan KNN 84,43% dengan nilai parameter *weights distance* dan *n_neighbors* 15 dan 20. Dengan membandingkan akurasi terbaik dari masing-masing algoritma maka dapat disimpulkan bahwa akurasi terbaik didapatkan pada algoritma SVM *kernel polynomial* dengan nilai parameter *degree* 9 dan *C* 2.5 yaitu 85.71%.

5.2. Saran

Dengan selesainya pembuatan sistem pada penelitian ini serta pelaporannya, dalam hal ini skripsi. Penulis menyadari bahwa masih banyak kekurangan yang terdapat pada sistem yang telah dibuat. Oleh karena itu, penulis bermaksud untuk menyampaikan beberapa saran dalam pengembangan sistem selanjutnya.

1. Dataset yang digunakan pada penelitian berasal dari *Kaggle dataset* dengan jumlah fitur sebenarnya yaitu 30 fitur, namun pada penelitian ini hanya digunakan 16 fitur dikarenakan keterbatasan pengetahuan penulis dalam melakukan ekstraksi fitur pada setiap fitur yang terdapat di dalam dataset. Oleh karena itu, untuk pengembangan selanjutnya disarankan agar dapat dikembangkan dengan menggunakan keseluruhan fitur secara mkasimal.
2. Terkait dengan *algoritma machine learning* yang digunakan, disarankan agar dapat mencoba dengan menggunakan *algoritma machine learning* yang lain sehingga dapat meningkatkan akurasi dari sistem.

3. Optimasi parameter yang digunakan pada penelitian ini hanya dengan menggunakan metode coba-coba yang artinya bahwa nilai parameter yang diubah-ubah untuk optimasi parameter murni merupakan nilai yang ditentukan oleh penulis sendiri dengan melakukan beberapa studi literatur. Oleh karena itu, disarankan agar dapat menggunakan metode yang lebih baik untuk meningkatkan hasil optimasi parameter yang dilakukan.
4. Sistem deteksi phishing dibuat dalam bentuk *Command Line Interface* (CLI). Oleh karena itu, disarankan agar kedepannya sistem dapat dikembangkan dengan dibuat dalam bentuk yang lebih baik misalnya dengan berbasis web, desktop, ataupun mobile.

DAFTAR PUSTAKA

Hakim, Zainal Arifin Al. 2016. Cyber Crime Dalam Bentuk Phising Dalam Undang-Undang Nomor 11 Tahun 2008 Tentang Informasi Dan Transaksi Elektronik Perspektif Hukum Pidana Islam. Surabaya: Universitas Islam Negeri Sunan Ampel.

Natan, Oskar dkk. 2019. "Grid SVM: Aplikasi Machine Learning dalam Pengolahan Data Akuakultur". Jurnal Rekayasa Elektrika Volume 15 (hlm 7-17). Surabaya: Politeknik Elektronika Negeri Surabaya.

Karima, Inna Sabily. 2014. Optimasi Parameter Pada Support Vector Machine untuk Klasifikasi Fragmen Metagenome Menggunakan Algoritme Genetika. Bogor: Institut Pertanian Bogor.

Halim Z. 2017. "Prediksi Website Pemancing Informasi Penting Phising Menggunakan Support Vector Machine (SVM)". Information System for Educators and Professionals. 2 (1): 71 – 82

Diani, Rima dkk. 2017. "Analisis Pengaruh Kernel Support Vector Machine (SVM) pada Klasifikasi Data Microarray untuk Deteksi Kanker". IndonesiaJournal on Computing Volume 2 (hlm. 109-118).

Putra, Jan Wira Gotama. 2019. Pengenalan Konsep Pembelajaran Mesin dan Deep Learning.Tokyo: Tokyo Institute of Technology.

ıad, Rami M dkk. "Phishing Websites Features".

Purwiantono, Febry Eka, dan Aris, Tjahyanto. 2017. “Model Klasifikasi untuk Deteksi Situs Phising di Indonesia”. Surabaya: Institut Teknologi Sepuluh Nopember.

Nugroho, Anto Satriyo. 2007. “Pengantar Support Vector Machine”. Jurnal Teknologi.

Erton, M Wiby. 2018. “Support Vector Machine”. Jombang: Universitas KH. A. Wahab Hasbullah.

Mti.binus.ac.id. (2017, 24 November). 10 Fold-Cross Validation. Diakses pada 12 November 2019, dari https://mti.binus.ac.id/2017/11/24/10-fold-cross-validation

Stats.stackexchange.com. (2012, 23 Juni). What is the influence of C in SVMs with linear kernel?. Diakses pada 12 November 2019, dari https://stats.stackexchange.com/questions/31066/what-is-the-influence-of-c-in-svms-with-linear-kernel

Towardsdatascience.com. (2019, 1 Agustus). Understanding Decision Trees for Classification (Python). Diakses pada 12 November 2019, dari https://towardsdatascience.com/understanding-decision-trees-for-classification-python-9663d683c952

Advernesia.com. (2019, 1 Mei). Apa itu Machine Learning dan Cara Kerjanya. iakses pada 10 November 2019, dari tps://www.advernesia.com/blog/data-science/machine-learning-adalah/

LAMPIRAN

Lampiran 1. *Confusion Matrix* SVM *linear*

Iterasi	C	TP	FP	FN	TN	Akurasi
1	1,0	363	118	126	499	77,94%
	1,5	364	117	130	495	77,67%
	2,5	365	116	128	497	77,94%
	5,0	363	118	130	495	77,58%
2	1,0	357	118	168	463	74,14%
	1,5	357	118	168	463	74,14%
	2,5	357	118	168	463	74,14%
	5,0	357	118	168	463	74,14%
3	1,0	372	101	145	488	77,76%
	1,5	372	101	145	488	77,76%
	2,5	372	101	145	488	77,76%
	5,0	372	101	145	488	77,76%
4	1,0	365	113	136	492	77,49%
	1,5	365	113	136	492	77,49%
	2,5	365	113	136	492	77,49%
	5,0	365	113	136	492	77,49%
5	1,0	382	111	141	472	77,22%
	1,5	382	111	141	472	77,22%
	2,5	382	111	141	472	77,22%
	5,0	382	111	143	470	77,03%
6	1,0	364	139	97	505	78,64%
	1,5	368	135	105	497	78,28%
	2,5	367	136	106	496	78,10%
	5,0	368	135	105	497	78,28%
7	1,0	371	123	145	466	75,75%
	1,5	371	123	145	466	75,75%
	2,5	371	123	145	466	75,75%
	5,0	371	123	145	466	75,75%
8	1,0	365	126	128	486	77,01%
	1,5	364	127	128	486	76,92%
	2,5	365	126	128	486	77,01%
	5,0	366	125	129	485	77,01%
	1,0	382	128	131	464	76,56%
	1,5	382	128	131	464	76,56%
	2,5	382	128	131	464	76,56%
	5,0	382	128	131	464	76,56%
0	1,0	381	119	120	485	78,37%
	1,5	381	119	120	485	78,37%

	2,5	381	119	120	485	78,37%
	5,0	381	119	120	485	78,37%

C	Akurasi
1,0	77,09%
1,5	77,01%
2,5	77,03%
5,0	77,00%

Lampiran 2. *Confusion Matrix* SVM *kernel polynomial*

***degree* = 1**

Iterasi	degree	C	TP	FP	FN	TN	Akurasi
1	1	1,0	364	117	128	497	77,85%
		1,5	364	117	128	497	77,85%
		2,5	365	116	130	495	77,76%
		5,0	363	118	128	497	77,76%
2	1	1,0	357	118	168	463	74,14%
		1,5	357	118	168	463	74,14%
		2,5	357	118	168	463	74,14%
		5,0	357	118	168	463	74,14%
3	1	1,0	373	100	151	482	77,31%
		1,5	373	100	153	480	77,12%
		2,5	373	100	152	481	77,22%
		5,0	372	101	148	485	77,49%
4	1	1,0	367	111	138	490	77,49%
		1,5	369	109	138	490	77,67%
		2,5	368	110	139	489	77,49%
		5,0	368	110	138	490	77,58%
5	1	1,0	381	112	143	470	76,94%
		1,5	382	111	141	472	77,22%
		2,5	382	111	143	470	77,03%
		5,0	382	111	141	472	77,22%
6	1	1,0	369	134	104	498	78,46%
		1,5	369	134	104	498	78,46%
		2,5	369	134	104	498	78,46%
		5,0	368	135	104	498	78,37%
7	1	1,0	371	123	145	466	75,75%
		1,5	371	123	145	466	75,75%
		2,5	371	123	145	466	75,75%
		5,0	371	123	145	466	75,75%
8	1	1,0	366	125	128	486	77,10%
		1,5	367	124	129	485	77,10%
		2,5	367	124	128	486	77,19%
		5,0	366	125	128	486	77,10%
	1	1,0	382	128	131	464	76,56%
		1,5	382	128	131	464	76,56%
		2,5	382	128	131	464	76,56%
		5,0	382	128	131	464	76,56%

10	1	1,0	381	119	120	485	78,37%
		1,5	381	119	120	485	78,37%
		2,5	381	119	120	485	78,37%
		5,0	381	119	120	485	78,37%

degree	C	Akurasi
1	1,0	77,00%
	1,5	77,02%
	2,5	77,00%
	5,0	77,03%

***degree* = 3**

Iterasi	degree	C	TP	FP	FN	TN	Akurasi
1	3	1,0	356	125	63	562	83,00%
		1,5	363	118	65	560	83,45%
		2,5	367	114	65	560	83,82%
		5,0	373	108	68	557	84,09%
2	3	1,0	376	99	95	536	82,46%
		1,5	384	91	96	535	83,09%
		2,5	384	91	93	538	83,36%
		5,0	377	98	84	547	83,54%
3	3	1,0	379	94	85	548	83,82%
		1,5	382	91	82	551	84,36%
		2,5	383	90	79	554	84,72%
		5,0	377	96	80	553	84,09%
4	3	1,0	378	100	90	538	82,82%
		1,5	377	101	86	542	83,09%
		2,5	381	97	92	536	82,91%
		5,0	382	96	94	534	82,82%
5	3	1,0	370	123	79	534	81,74%
		1,5	368	125	80	533	81,46%
		2,5	370	123	79	534	81,74%
		5,0	378	115	76	537	82,73%
	3	1,0	378	125	61	541	83,17%
		1,5	385	118	61	541	83,80%
		2,5	386	117	64	538	83,62%
		5,0	387	116	69	533	83,26%
	3	1,0	376	118	77	534	82,35%
		1,5	381	113	74	537	83,08%

		2,5	383	111	75	536	83,17%
		5,0	393	101	77	534	83,89%
8	3	1,0	384	107	69	545	84,07%
		1,5	389	102	66	548	84,80%
		2,5	390	101	69	545	84,62%
		5,0	393	98	70	544	84,80%
9	3	1,0	418	92	66	529	85,70%
		1,5	418	92	67	528	85,61%
		2,5	420	90	69	526	85,61%
		5,0	425	85	71	524	85,88%
10	3	1,0	382	118	67	538	83,26%
		1,5	384	116	68	537	83,35%
		2,5	394	106	75	530	83,62%
		5,0	396	104	75	530	83,80%

degree	C	Akurasi
3	1,0	83,24%
	1,5	83,61%
	2,5	83,72%
	5,0	83,89%

***degree* = 5**

Iterasi	degree	C	TP	FP	FN	TN	Akurasi
1	5	1,0	372	109	60	565	84,72%
		1,5	375	106	65	560	84,54%
		2,5	379	102	67	558	84,72%
		5,0	381	100	67	558	84,90%
2	5	1,0	384	91	77	554	84,81%
		1,5	385	90	81	550	84,54%
		2,5	388	87	80	551	84,90%
		5,0	395	80	79	552	85,62%
3	5	1,0	389	84	79	554	85,26%
		1,5	388	85	78	555	85,26%
		2,5	392	81	81	552	85,35%
		5,0	395	78	87	546	85,08%
	5	1,0	380	98	80	548	83,91%
		1,5	380	98	77	551	84,18%
		2,5	383	95	78	550	84,36%
		5,0	395	83	84	544	84,90%

5	5	1,0	382	111	79	534	82,82%
		1,5	381	112	77	536	82,91%
		2,5	380	113	73	540	83,18%
		5,0	383	110	71	542	83,63%
6	5	1,0	380	123	65	537	82,99%
		1,5	380	123	66	536	82,90%
		2,5	385	118	68	534	83,17%
		5,0	394	109	70	532	83,80%
7	5	1,0	397	97	78	533	84,16%
		1,5	404	90	78	533	84,80%
		2,5	402	92	81	530	84,34%
		5,0	402	92	79	532	84,52%
8	5	1,0	393	98	72	542	84,62%
		1,5	397	94	70	544	85,16%
		2,5	398	93	69	545	85,34%
		5,0	396	95	74	540	84,71%
9	5	1,0	430	80	62	533	87,15%
		1,5	432	78	61	534	87,42%
		2,5	434	76	65	530	87,24%
		5,0	436	74	64	531	87,51%
10	5	1,0	397	103	62	543	85,07%
		1,5	392	108	62	543	84,62%
		2,5	394	106	57	548	85,25%
		5,0	404	96	62	543	85,70%

degree	C	Akurasi
5	1,0	84,55%
	1,5	84,63%
	2,5	84,79%
	5,0	85,04%

***degree* = 7**

Iterasi	degree	C	TP	FP	FN	TN	Akurasi
	7	1,0	397	84	99	526	83,45%
		1,5	387	94	66	559	85,53%
		2,5	389	92	71	554	85,26%
		5,0	392	89	72	553	85,44%
	7	1,0	400	75	118	513	82,55%
		1,5	397	78	82	549	85,53%

		2,5	397	78	83	548	85,44%
		5,0	399	76	82	549	85,71%
3	7	1,0	406	67	98	535	85,08%
		1,5	396	77	86	547	85,26%
		2,5	401	72	84	549	85,90%
		5,0	400	73	83	550	85,90%
4	7	1,0	399	79	101	527	83,73%
		1,5	398	80	87	541	84,90%
		2,5	401	77	89	539	84,99%
		5,0	399	79	83	545	85,35%
5	7	1,0	391	102	98	515	81,92%
		1,5	380	113	72	541	83,27%
		2,5	384	109	70	543	83,82%
		5,0	388	105	73	540	83,91%
6	7	1,0	411	92	87	515	83,80%
		1,5	408	95	74	528	84,71%
		2,5	411	92	74	528	84,98%
		5,0	408	95	66	536	85,43%
7	7	1,0	411	83	97	514	83,71%
		1,5	400	94	80	531	84,25%
		2,5	400	94	79	532	84,34%
		5,0	405	89	79	532	84,80%
8	7	1,0	411	80	93	521	84,34%
		1,5	399	92	71	543	85,25%
		2,5	407	84	71	543	85,97%
		5,0	408	83	68	546	86,33%
9	7	1,0	445	65	94	501	85,61%
		1,5	440	70	66	529	87,69%
		2,5	444	66	69	526	87,78%
		5,0	447	63	70	525	87,96%
10	7	1,0	411	89	86	519	84,16%
		1,5	412	88	71	534	85,61%
		2,5	405	95	69	536	85,16%
		5,0	407	93	65	540	85,70%

degree	C	Akurasi
	1,0	83,84%
	1,5	85,20%
	2,5	85,36%
	5,0	85,65%

***degree* = 9**

Iterasi	degree	C	TP	FP	FN	TN	Akurasi
1	9	1,0	390	91	71	554	85,35%
		1,5	392	89	72	553	85,44%
		2,5	393	88	70	555	85,71%
		5,0	392	89	71	554	85,53%
2	9	1,0	399	76	87	544	85,26%
		1,5	399	76	84	547	85,53%
		2,5	400	75	81	550	85,90%
		5,0	400	75	81	550	85,90%
3	9	1,0	401	72	84	549	85,90%
		1,5	401	72	84	549	85,90%
		2,5	401	72	82	551	86,08%
		5,0	401	72	82	551	86,08%
4	9	1,0	399	79	83	545	85,35%
		1,5	400	78	84	544	85,35%
		2,5	399	79	82	546	85,44%
		5,0	396	82	79	549	85,44%
5	9	1,0	389	104	73	540	84,00%
		1,5	389	104	74	539	83,91%
		2,5	386	107	73	540	83,73%
		5,0	385	108	72	541	83,73%
6	9	1,0	407	96	72	530	84,80%
		1,5	407	96	70	532	84,98%
		2,5	406	97	67	535	85,16%
		5,0	406	97	66	536	85,25%
7	9	1,0	404	90	80	531	84,62%
		1,5	405	89	79	532	84,80%
		2,5	405	89	79	532	84,80%
		5,0	404	90	76	535	84,98%
8	9	1,0	407	84	72	542	85,88%
		1,5	407	84	68	546	86,24%
		2,5	407	84	67	547	86,33%
		5,0	406	85	67	547	86,24%
	9	1,0	445	65	71	524	87,69%
		1,5	446	64	69	526	87,96%
		2,5	447	63	69	526	88,05%
		5,0	445	65	70	525	87,78%
	9	1,0	412	88	69	536	85,79%
		1,5	406	94	70	535	85,16%

		2,5	408	92	64	541	85,88%
		5,0	408	92	64	541	85,88%

degree	C	Akurasi
9	1,0	85,46%
	1,5	85,53%
	2,5	85,71%
	5,0	85,68%

Lampiran 3. *Confusion Matrix* SVM *kernel* RBF

***gamma* = 0.1**

Iterasi	gamma	C	TP	FP	FN	TN	Akurasi
1	0,1	1,0	374	107	59	566	84,99%
		1,5	371	110	64	561	84,27%
		2,5	371	110	61	564	84,54%
		5,0	376	105	65	560	84,63%
2	0,1	1,0	375	100	79	552	83,82%
		1,5	378	97	78	553	84,18%
		2,5	382	93	79	552	84,45%
		5,0	381	94	82	549	84,09%
3	0,1	1,0	386	87	74	559	85,44%
		1,5	389	84	77	556	85,44%
		2,5	388	85	79	554	85,17%
		5,0	387	86	81	552	84,90%
4	0,1	1,0	376	102	80	548	83,54%
		1,5	378	100	76	552	84,09%
		2,5	381	97	77	551	84,27%
		5,0	381	97	79	549	84,09%
5	0,1	1,0	369	124	78	535	81,74%
		1,5	370	123	75	538	82,10%
		2,5	371	122	68	545	82,82%
		5,0	376	117	73	540	82,82%
6	0,1	1,0	378	125	63	539	82,99%
		1,5	379	124	62	540	83,17%
		2,5	380	123	63	539	83,17%
		5,0	378	125	63	539	82,99%
7	0,1	1,0	381	113	71	540	83,35%
		1,5	382	112	69	542	83,62%
		2,5	384	110	67	544	83,98%
		5,0	391	103	74	537	83,98%
8	0,1	1,0	388	103	65	549	84,80%
		1,5	389	102	63	551	85,07%
		2,5	388	103	67	547	84,62%
		5,0	392	99	70	544	84,71%
	0,1	1,0	423	87	64	531	86,33%
		1,5	425	85	61	534	86,79%
		2,5	423	87	57	538	86,97%
		5,0	432	78	61	534	87,42%

10	0,1	1,0	393	107	65	540	84,43%
		1,5	391	109	61	544	84,62%
		2,5	390	110	59	546	84,71%
		5,0	391	109	58	547	84,89%

gamma	C	Akurasi
0,1	1,0	84,14%
	1,5	84,33%
	2,5	84,47%
	5,0	84,45%

***gamma* = 0.5**

Iterasi	gamma	C	TP	FP	FN	TN	Akurasi
1	0,5	1,0	389	92	67	558	85,62%
		1,5	390	91	68	557	85,62%
		2,5	392	89	67	558	85,90%
		5,0	392	89	66	559	85,99%
2	0,5	1,0	396	79	81	550	85,53%
		1,5	395	80	80	551	85,53%
		2,5	394	81	78	553	85,62%
		5,0	391	84	80	551	85,17%
3	0,5	1,0	393	80	81	552	85,44%
		1,5	394	79	81	552	85,53%
		2,5	392	81	81	552	85,35%
		5,0	392	81	77	556	85,71%
4	0,5	1,0	391	87	82	546	84,72%
		1,5	392	86	79	549	85,08%
		2,5	390	88	76	552	85,17%
		5,0	389	89	77	551	84,99%
5	0,5	1,0	380	113	70	543	83,45%
		1,5	383	110	71	542	83,63%
		2,5	380	113	70	543	83,45%
		5,0	378	115	71	542	83,18%
	0,5	1,0	396	107	66	536	84,34%
		1,5	400	103	64	538	84,89%
		2,5	401	102	64	538	84,98%

		5,0	401	102	63	539	85,07%
7	0,5	1,0	400	94	79	532	84,34%
		1,5	401	93	78	533	84,52%
		2,5	399	95	75	536	84,62%
		5,0	400	94	77	534	84,52%
8	0,5	1,0	400	91	68	546	85,61%
		1,5	402	89	68	546	85,79%
		2,5	402	89	65	549	86,06%
		5,0	401	90	65	549	85,97%
9	0,5	1,0	441	69	69	526	87,51%
		1,5	440	70	69	526	87,42%
		2,5	438	72	68	527	87,33%
		5,0	437	73	67	528	87,33%
10	0,5	1,0	404	96	68	537	85,16%
		1,5	399	101	66	539	84,89%
		2,5	397	103	66	539	84,71%
		5,0	397	103	66	539	84,71%

gamma	C	Akurasi
0,5	1,0	85,17%
	1,5	85,29%
	2,5	85,32%
	5,0	85,26%

***gamma* = 1.0**

Iterasi	gamma	C	TP	FP	FN	TN	Akurasi
1	1,0	1,0	386	95	63	562	85,71%
		1,5	386	95	63	562	85,71%
		2,5	386	95	63	562	85,71%
		5,0	386	95	63	562	85,71%
[illegible]	1,0	1,0	388	87	81	550	84,81%
		1,5	385	90	79	552	84,72%
		2,5	385	90	79	552	84,72%
		5,0	385	90	79	552	84,72%
[illegible]	1,0	1,0	382	91	78	555	84,72%
		1,5	382	91	78	555	84,72%

		2,5	382	91	78	555	84,72%
		5,0	382	91	78	555	84,72%
4	1,0	1,0	379	99	77	551	84,09%
		1,5	378	100	76	552	84,09%
		2,5	378	100	76	552	84,09%
		5,0	378	100	76	552	84,09%
5	1,0	1,0	368	125	72	541	82,19%
		1,5	367	126	72	541	82,10%
		2,5	367	126	72	541	82,10%
		5,0	367	126	72	541	82,10%
6	1,0	1,0	389	114	63	539	83,98%
		1,5	391	112	63	539	84,16%
		2,5	391	112	63	539	84,16%
		5,0	391	112	63	539	84,16%
7	1,0	1,0	387	107	72	539	83,80%
		1,5	389	105	75	536	83,71%
		2,5	388	106	75	536	83,62%
		5,0	388	106	75	536	83,62%
8	1,0	1,0	391	100	67	547	84,89%
		1,5	392	99	67	547	84,98%
		2,5	393	98	67	547	85,07%
		5,0	393	98	67	547	85,07%
9	1,0	1,0	422	88	68	527	85,88%
		1,5	421	89	69	526	85,70%
		2,5	421	89	69	526	85,70%
		5,0	421	89	69	526	85,70%
10	1,0	1,0	383	117	64	541	83,62%
		1,5	383	117	65	540	83,53%
		2,5	383	117	65	540	83,53%
		5,0	383	117	65	540	83,53%

gamma	C	Akurasi
1,0	1,0	84,37%
	1,5	84,34%
	2,5	84,34%
	5,0	84,34%

***gamma* = 1.5**

Iterasi	gamma	C	TP	FP	FN	TN	Akurasi
1	1,5	1,0	385	96	62	563	85,71%
		1,5	386	95	63	562	85,71%
		2,5	386	95	63	562	85,71%
		5,0	386	95	63	562	85,71%
2	1,5	1,0	381	94	79	552	84,36%
		1,5	381	94	79	552	84,36%
		2,5	381	94	79	552	84,36%
		5,0	381	94	79	552	84,36%
3	1,5	1,0	377	96	77	556	84,36%
		1,5	379	94	77	556	84,54%
		2,5	379	94	77	556	84,54%
		5,0	379	94	77	556	84,54%
4	1,5	1,0	378	100	78	550	83,91%
		1,5	378	100	78	550	83,91%
		2,5	378	100	78	550	83,91%
		5,0	378	100	78	550	83,91%
5	1,5	1,0	363	130	70	543	81,92%
		1,5	365	128	70	543	82,10%
		2,5	365	128	70	543	82,10%
		5,0	365	128	70	543	82,10%
6	1,5	1,0	388	115	62	540	83,98%
		1,5	388	115	62	540	83,98%
		2,5	388	115	62	540	83,98%
		5,0	388	115	62	540	83,98%
7	1,5	1,0	388	106	74	537	83,71%
		1,5	387	107	74	537	83,62%
		2,5	387	107	74	537	83,62%
		5,0	387	107	74	537	83,62%
8	1,5	1,0	390	101	65	549	84,98%
		1,5	391	100	65	549	85,07%
		2,5	391	100	65	549	85,07%
		5,0	391	100	65	549	85,07%
	1,5	1,0	416	94	67	528	85,43%
		1,5	417	93	67	528	85,52%
		2,5	417	93	67	528	85,52%
		5,0	417	93	67	528	85,52%
	1,5	1,0	381	119	64	541	83,44%

		1,5	380	120	65	540	83,26%
		2,5	380	120	65	540	83,26%
		5,0	380	120	65	540	83,26%

gamma	C	Akurasi
1,5	1,0	84,18%
	1,5	84,21%
	2,5	84,21%
	5,0	84,21%

***gamma* = 2.0**

Iterasi	gamma	C	TP	FP	FN	TN	Akurasi
1	2,0	1,0	382	99	62	563	85,44%
		1,5	382	99	62	563	85,44%
		2,5	382	99	62	563	85,44%
		5,0	382	99	62	563	85,44%
2	2,0	1,0	378	97	78	553	84,18%
		1,5	378	97	78	553	84,18%
		2,5	378	97	78	553	84,18%
		5,0	378	97	78	553	84,18%
3	2,0	1,0	377	96	77	556	84,36%
		1,5	377	96	77	556	84,36%
		2,5	377	96	77	556	84,36%
		5,0	377	96	77	556	84,36%
4	2,0	1,0	376	102	78	550	83,73%
		1,5	377	101	78	550	83,82%
		2,5	377	101	78	550	83,82%
		5,0	377	101	78	550	83,82%
5	2,0	1,0	363	130	70	543	81,92%
		1,5	363	130	70	543	81,92%
		2,5	363	130	70	543	81,92%
		5,0	363	130	70	543	81,92%
	2,0	1,0	387	116	61	541	83,98%
		1,5	387	116	61	541	83,98%
		2,5	387	116	61	541	83,98%
		5,0	387	116	61	541	83,98%

7	2,0	1,0	387	107	73	538	83,71%
		1,5	387	107	74	537	83,62%
		2,5	387	107	74	537	83,62%
		5,0	387	107	74	537	83,62%
8	2,0	1,0	386	105	63	551	84,80%
		1,5	386	105	63	551	84,80%
		2,5	386	105	63	551	84,80%
		5,0	386	105	63	551	84,80%
9	2,0	1,0	415	95	66	529	85,43%
		1,5	415	95	67	528	85,34%
		2,5	415	95	67	528	85,34%
		5,0	415	95	67	528	85,34%
10	2,0	1,0	379	121	64	541	83,26%
		1,5	379	121	64	541	83,26%
		2,5	379	121	64	541	83,26%
		5,0	379	121	64	541	83,26%

gamma	C	Akurasi
2,0	1,0	84,08%
	1,5	84,07%
	2,5	84,07%
	5,0	84,07%

Lampiran 4. *Confusion Matrix Decision Tree*

criterion = gini

Iterasi	criterion	max_depth	TP	FP	FN	TN	Akurasi
1	gini	1	328	153	161	464	71,61%
		5	327	154	76	549	79,20%
		10	364	117	74	551	82,73%
		15	385	96	70	555	84,99%
		20	386	95	69	556	85,17%
2	gini	1	329	146	194	437	69,26%
		5	360	115	114	517	79,29%
		10	383	92	92	539	83,36%
		15	394	81	83	548	85,17%
		20	395	80	85	546	85,08%
3	gini	1	336	137	197	436	69,80%
		5	345	128	78	555	81,37%
		10	389	84	79	554	85,26%
		15	399	74	81	552	85,99%
		20	400	73	81	552	86,08%
4	gini	1	341	137	191	437	70,34%
		5	331	147	82	546	79,29%
		10	377	101	94	534	82,37%
		15	391	87	83	545	84,63%
		20	392	86	82	546	84,81%
5	gini	1	343	150	199	414	68,44%
		5	336	157	77	536	78,84%
		10	382	111	74	539	83,27%
		15	386	107	80	533	83,09%
		20	387	106	79	534	83,27%
6	gini	1	325	178	184	418	67,24%
		5	336	167	66	536	78,91%
		10	382	121	70	532	82,71%
		15	403	100	67	535	84,89%
		20	401	102	66	536	84,80%
	gini	1	346	148	180	431	70,32%
		5	349	145	81	530	79,55%
		10	391	103	82	529	83,26%
		15	405	89	78	533	84,89%
		20	405	89	79	532	84,80%

8	gini	1	337	154	174	440	70,32%
		5	334	157	72	542	79,28%
		10	388	103	96	518	81,99%
		15	407	84	69	545	86,15%
		20	408	83	76	538	85,61%
9	gini	1	355	155	188	407	68,96%
		5	381	129	65	530	82,44%
		10	432	78	83	512	85,43%
		15	445	65	74	521	87,42%
		20	444	66	75	520	87,24%
10	gini	1	351	149	165	440	71,58%
		5	357	143	64	541	81,27%
		10	390	110	64	541	84,25%
		15	404	96	68	537	85,16%
		20	406	94	69	536	85,25%

criterion	max_depth	Akurasi
gini	1	69,79%
	5	79,95%
	10	83,46%
	15	85,24%
	20	85,21%

criterion = entropy

Iterasi	criterion	max_depth	TP	FP	FN	TN	Akurasi
1	entropy	1	481	0	476	149	56,96%
		5	327	154	76	549	79,20%
		10	363	118	69	556	83,09%
		15	384	97	67	558	85,17%
		20	385	96	69	556	85,08%
	entropy	1	475	0	502	129	54,61%
		5	360	115	114	517	79,29%
		10	376	99	84	547	83,45%
		15	398	77	82	549	85,62%
		20	397	78	84	547	85,35%
	entropy	1	473	0	475	158	57,05%

		5	344	129	77	556	81,37%
		10	392	81	84	549	85,08%
		15	400	73	82	551	85,99%
		20	400	73	82	551	85,99%
4	entropy	1	478	0	470	158	57,50%
		5	331	147	82	546	79,29%
		10	377	101	94	534	82,37%
		15	390	88	82	546	84,63%
		20	391	87	82	546	84,72%
5	entropy	1	493	0	460	153	58,41%
		5	336	157	77	536	78,84%
		10	371	122	65	548	83,09%
		15	389	104	79	534	83,45%
		20	388	105	77	536	83,54%
6	entropy	1	503	0	451	151	59,19%
		5	336	167	66	536	78,91%
		10	384	119	69	533	82,99%
		15	399	104	62	540	84,98%
		20	401	102	66	536	84,80%
7	entropy	1	494	0	470	141	57,47%
		5	349	145	81	530	79,55%
		10	384	110	81	530	82,71%
		15	405	89	79	532	84,80%
		20	405	89	80	531	84,71%
8	entropy	1	491	0	468	146	57,65%
		5	334	157	72	542	79,28%
		10	372	119	84	530	81,63%
		15	405	86	65	549	86,33%
		20	406	85	74	540	85,61%
9	entropy	1	510	0	459	136	58,46%
		5	381	129	65	530	82,44%
		10	423	87	62	533	86,52%
		15	442	68	72	523	87,33%
		20	442	68	74	521	87,15%
10	entropy	1	500	0	461	144	58,28%
		5	357	143	64	541	81,27%
		10	385	115	72	533	83,08%
		15	407	93	65	540	85,70%
		20	404	96	68	537	85,16%

criterion	max_depth	Akurasi
entropy	1	57,56%
	5	79,95%
	10	83,40%
	15	85,40%
	20	85,21%

Lampiran 5. *Confusion Matrix K-Nearest Neighbor*

weights = uniform

Iterasi	weights	n_neighbors	TP	FP	FN	TN	Akurasi
1	uniform	1	376	105	118	507	79,84%
		5	372	109	91	534	81,92%
		10	404	77	118	507	82,37%
		15	384	97	92	533	82,91%
		20	391	90	104	521	82,46%
2	uniform	1	382	93	123	508	80,47%
		5	388	87	104	527	82,73%
		10	412	63	144	487	81,28%
		15	374	101	98	533	82,01%
		20	387	88	114	517	81,74%
3	uniform	1	369	104	93	540	82,19%
		5	371	102	106	527	81,19%
		10	395	78	134	499	80,83%
		15	380	93	116	517	81,10%
		20	386	87	128	505	80,56%
4	uniform	1	391	87	112	516	82,01%
		5	390	88	117	511	81,46%
		10	399	79	137	491	80,47%
		15	390	88	114	514	81,74%
		20	395	83	109	519	82,64%
5	uniform	1	354	139	77	536	80,47%
		5	363	130	88	525	80,29%
		10	378	115	115	498	79,20%
		15	373	120	107	506	79,48%
		20	383	110	112	501	79,93%
6	uniform	1	355	148	109	493	76,74%
		5	366	137	78	524	80,54%
		10	401	102	107	495	81,09%
		15	384	119	96	506	80,54%
		20	393	110	101	501	80,90%
	uniform	1	385	109	123	488	79,00%
		5	384	110	114	497	79,73%
		10	394	100	140	471	78,28%
		15	388	106	116	495	79,91%
		20	395	99	115	496	80,63%

8	uniform	1	366	125	95	519	80,09%
		5	387	104	91	523	82,35%
		10	405	86	116	498	81,72%
		15	383	108	89	525	82,17%
		20	389	102	98	516	81,90%
9	uniform	1	388	122	100	495	79,91%
		5	428	82	90	505	84,43%
		10	431	79	102	493	83,62%
		15	430	80	95	500	84,16%
		20	434	76	100	495	84,07%
10	uniform	1	378	122	83	522	81,45%
		5	399	101	83	522	83,35%
		10	420	80	106	499	83,17%
		15	388	112	81	524	82,53%
		20	405	95	106	499	81,81%

weights	n_neighbors	Akurasi
uniform	1	80,22%
	5	81,80%
	10	81,20%
	15	81,66%
	20	81,66%

weights = distance

Iterasi	weights	n_neighbors	TP	FP	FN	TN	Akurasi
1	distance	1	376	105	118	507	79,84%
		5	378	103	83	542	83,18%
		10	409	72	96	529	84,81%
		15	399	82	83	542	85,08%
		20	398	83	84	541	84,90%
	distance	1	382	93	123	508	80,47%
		5	394	81	93	538	84,27%
		10	413	62	113	518	84,18%
		15	384	91	89	542	83,73%
		20	393	82	93	538	84,18%
	distance	1	369	104	93	540	82,19%

		5	384	89	94	539	83,45%
		10	404	69	104	529	84,36%
		15	403	70	94	539	85,17%
		20	402	71	97	536	84,81%
4	distance	1	391	87	112	516	82,01%
		5	391	87	98	530	83,27%
		10	406	72	104	524	84,09%
		15	403	75	102	526	84,00%
		20	405	73	102	526	84,18%
5	distance	1	354	139	77	536	80,47%
		5	374	119	70	543	82,91%
		10	388	105	86	527	82,73%
		15	389	104	85	528	82,91%
		20	392	101	87	526	83,00%
6	distance	1	355	148	109	493	76,74%
		5	376	127	68	534	82,35%
		10	409	94	91	511	83,26%
		15	404	99	83	519	83,53%
		20	411	92	88	514	83,71%
7	distance	1	385	109	123	488	79,00%
		5	389	105	95	516	81,90%
		10	405	89	106	505	82,35%
		15	408	86	95	516	83,62%
		20	411	83	95	516	83,89%
8	distance	1	366	125	95	519	80,09%
		5	393	98	87	527	83,26%
		10	409	82	88	526	84,62%
		15	398	93	76	538	84,71%
		20	404	87	83	531	84,62%
9	distance	1	388	122	100	495	79,91%
		5	434	76	80	515	85,88%
		10	441	69	84	511	86,15%
		15	446	64	82	513	86,79%
		20	447	63	88	507	86,33%
10	distance	1	378	122	83	522	81,45%
		5	402	98	74	531	84,43%
		10	415	85	81	524	84,98%
		15	402	98	70	535	84,80%
		20	413	87	82	523	84,71%

weights	n_neighbors	Akurasi
distance	1	80,22%
	5	83,49%
	10	84,15%
	15	84,43%
	20	84,43%

Lampiran 6. *Source Code Program*

evaluasi.py

```
import pandas as pd
import numpy as np
from sklearn.model_selection import KFold
from sklearn.preprocessing import MinMaxScaler
from sklearn.svm import SVC
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score
from sklearn.metrics import confusion_matrix
from sklearn.metrics import classification_report
from termcolor import colored


dataset = pd.read_csv('data_training.csv')

x = dataset.drop('Result', axis=1)
y = dataset['Result']

scaler = MinMaxScaler(feature_range=(-1, 1))
x = scaler.fit_transform(x)

scoresl = []
scoresp = []
scoresr = []
scoresDT = []
scoresKNN = []

cv = KFold(n_splits=10, random_state=42, shuffle=True)

print(colored('\n SUPPORT VECTOR MACHINE (SVM)',
attrs=['bold']))
print(colored(' Linear:', attrs=['bold']))
cl = input(' C  : ')
cl_input = float(cl)
cSVMl = SVC(kernel='linear', C=cl_input)
print(colored(' ----------------------------------------
--------', attrs=['bold']))
for train_index, test_index in cv.split(x):
    print(" Train       : ", train_index)
    print(" Test  : ", test_index)

    x_train, x_test, y_train, y_test = x[train_index],
x[test_index], y[train_index], y[test_index]
    cSVMl.fit(x_train, y_train)
    Y_SVMl = cSVMl.predict(x_test)
    skorl = cSVMl.score(x_test, y_test)*100
    print(" Score       :  [%.2f" % skorl, "%]")
    scoresl.append(skorl)
    cmlinear = confusion_matrix(y_test, Y_SVMl)
    print('', cmlinear)

urasi_SVMl = np.mean(scoresl)
```

```
print(" Akurasi:  [%.2f" % akurasi_SVMl, "%]")

print(colored('\n Polynomial:', attrs=['bold']))
degree = input(' Degree : ')
degree_input = float(degree)
cp = input(' C    : ')
cp_input = float(cp)
cSVMp = SVC(kernel='poly', gamma='scale',
degree=degree_input, C=cp_input)
print(colored(' ----------------------------------------
--------', attrs=['bold']))
for train_index, test_index in cv.split(x):
    print(" Train       : ", train_index)
    print(" Test  : ", test_index)

    x_train, x_test, y_train, y_test = x[train_index],
x[test_index], y[train_index], y[test_index]
    cSVMp.fit(x_train, y_train)
    Y_SVMp = cSVMp.predict(x_test)
    skorp = cSVMp.score(x_test, y_test)*100
    print(" Score       :  [%.2f" % skorp, "%]")
    scoresp.append(skorp)
    cmpoly = confusion_matrix(y_test, Y_SVMp)
    print('', cmpoly)

akurasi_SVMp = np.mean(scoresp)
print(" Akurasi:  [%.2f" % akurasi_SVMp, "%]")

print(colored('\n RBF:', attrs=['bold']))
gamma = input(' Gamma   : ')
gamma_input = float(gamma)
cr = input(' C    : ')
cr_input = float(cr)
cSVMr = SVC(kernel='rbf', gamma=gamma_input, C=cr_input)
print(colored(' ----------------------------------------
--------', attrs=['bold']))
for train_index, test_index in cv.split(x):
    print(" Train       : ", train_index)
    print(" Test  : ", test_index)

    x_train, x_test, y_train, y_test = x[train_index],
x[test_index], y[train_index], y[test_index]
    cSVMr.fit(x_train, y_train)
    Y_SVMr = cSVMr.predict(x_test)
    skorr = cSVMr.score(x_test, y_test)*100
    print(" Score       :  [%.2f" % skorr, "%]")
    scoresr.append(skorr)
    cmrbf = confusion_matrix(y_test, Y_SVMr)
    print('', cmrbf)

urasi_SVMr = np.mean(scoresr)
int(" Akurasi:  [%.2f" % akurasi_SVMr, "%]")

int(colored('\n DECISION TREE', attrs=['bold']))
int(colored(' *criterion = gini / entropy', 'grey'))
int(colored(' *max depth = integer', 'grey'))
```

```
criterion = input(' Criterion   : ')
criterion_input = str(criterion)
max_depth = input(' Max Depth   : ')
max_depth_input = int(max_depth)
cDT = DecisionTreeClassifier(criterion=criterion_input,
max_depth=max_depth_input)
print(colored(' ------------------------------------------
--------', attrs=['bold']))
for train_index, test_index in cv.split(x):
    print(" Train   : ", train_index)
    print(" Test    : ", test_index)

    x_train, x_test, y_train, y_test = x[train_index],
x[test_index], y[train_index], y[test_index]
    cDT.fit(x_train, y_train)
    Y_DT = cDT.predict(x_test)
    skorDT = cDT.score(x_test, y_test)*100
    print(" Score   :  [%.2f" % skorDT, "%]")
    scoresDT.append(skorDT)
    cmDT = confusion_matrix(y_test, Y_DT)
    print('', cmDT)

akurasi_DT = np.mean(scoresDT)
print(" Akurasi:  [%.2f" % akurasi_DT, "%]")

print(colored('\n K-NEAREST NEIGHBORS', attrs=['bold']))
print(colored(' *weights     = uniform / distance',
'grey'))
print(colored(' *n_neighbors = integer', 'grey'))
weights = input(' Weights       : ')
weights_input = str(weights)
n_neighbors = input(' N Neighbors   : ')
n_neighbors_input = int(n_neighbors)
cKNN = KNeighborsClassifier(weights=weights_input,
n_neighbors=n_neighbors_input)
print(colored(' ------------------------------------------
--------', attrs=['bold']))
for train_index, test_index in cv.split(x):
    print(" Train   : ", train_index)
    print(" Test    : ", test_index)

    x_train, x_test, y_train, y_test = x[train_index],
x[test_index], y[train_index], y[test_index]
    cKNN.fit(x_train, y_train)
    Y_KNN = cKNN.predict(x_test)
    skorKNN = cKNN.score(x_test, y_test)*100
    print(" Score   :  [%.2f" % skorKNN, "%]")
    scoresKNN.append(skorKNN)
    cmKNN = confusion_matrix(y_test, Y_KNN)
    print('', cmKNN)

urasi_KNN = np.mean(scoresKNN)
int(" Akurasi:  [%.2f" % akurasi_KNN, "%]")

int(colored('\n AKURASI :', attrs=['bold']))
int(" Akurasi SVM : %.2f, %.2f, %.2f"
```

```
%(akurasi_SVMl,akurasi_SVMp,akurasi_SVMr), "%")
print(" Akurasi DT  : %.2f" % akurasi_DT, "%")
print(" Akurasi KNN : %.2f" % akurasi_KNN, "%")
```

make_model.py

```
import numpy as np
import pandas as pd
from sklearn.svm import SVC
import pickle

x = pd.read_csv('data_training.csv')
a = np.array(x)
y = a[:,16]

x = np.column_stack((x.having_IP_Address, x.URL_Length,
x.Shortining_Service, x.having_At_Symbol,
x.double_slash_redirecting, x.Prefix_Suffix,
x.having_Sub_Domain, x.Domain_registeration_length,
x.HTTPS_token, x.Submitting_to_email, x.RightClick,
x.Iframe, x.age_of_domain, x.DNSRecord, x.web_traffic,
x.Statistical_report))

cSVMl = SVC(kernel='linear')
cSVMl.fit(x, y)
model_namel = 'model_l.sav'
pickle.dump(cSVMl, open(model_namel, 'wb'))
print(' Model SVM Linear Successful')

cSVMp = SVC(kernel='poly', gamma='scale', degree=7, C=5.0)
cSVMp.fit(x, y)
model_namep = 'model_p.sav'
pickle.dump(cSVMp, open(model_namep, 'wb'))
print(' Model SVM Polynomial Successful')

cSVMr = SVC(kernel='rbf', gamma=0.3, C=5.0)
cSVMr.fit(x, y)
model_namer = 'model_r.sav'
pickle.dump(cSVMr, open(model_namer, 'wb'))
print(' Model SVM RBF Successful')
```

phishion.py

```
import whois
import dns.resolver
from datetime import datetime
om urllib.parse import urlparse
port urllib.request
port shutil
port re
port sys
port xmltodict
```

```
import json
import socket
import csv
from termcolor import colored

print(colored('          _     _     _     _              ',
'grey', attrs=['bold']))
print(colored('  ____ | |___(_)___| |___(_) ___  _____ ',
'grey', attrs=['bold']))
print(colored(' /  _ \|  _  | /  _)  _  | |/ _ \/  _  |',
'grey', attrs=['bold']))
print(colored(' | (_) | | | | | (_| | | | | (_) | | | |',
'grey', attrs=['bold']))
print(colored(' |  __/|_| |_|_|__ |_| |_| |\___/|_| |_|',
'grey', attrs=['bold']))
print(colored(' |_|-----------(___/-PHISHING-DETECTION-',
'grey', attrs=['bold']))

while(True):
      url = input(colored('\n INPUT URL: ', 'grey',
attrs=['bold']))
      print(colored('
=====================================', 'grey',
attrs=['bold']))
      regex = re.compile(
        r'^(?:http|ftp)s?://' # http:// or https://
        r'(?:(?:[A-Z0-9](?:[A-Z0-9-]{0,61}[A-Z0-
9])?\.)+(?:[A-Z]{2,6}\.?|[A-Z0-9-]{2,}\.?)|' #domain...
        r'localhost|' #localhost...
        r'\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3})' # ...or ip
        r'(?::\d+)?' # optional port
        r'(?:/?|[/?]\S+)$', re.IGNORECASE)
      check_url = re.match(regex, url) is not None

      if check_url == True:
            break
      else:
            print(colored(' Your Input is Not URL', 'red'))

print(colored(' 1. LINEAR', 'grey'))
print(colored(' 2. POLYNOMIAL', 'grey'))
print(colored(' 3. RBF', 'grey'))
pilih_kernel = input(colored(' Pilih Kernel     : ',
'grey', attrs=['bold']))
kernel = int(pilih_kernel)
print(colored(' =====================================',
'grey', attrs=['bold']))

t1 = datetime.now()
obj = urlparse(url)
stname = obj.hostname
y:
    pywhois = whois.whois(hostname)
cept:
    next
```

```
reader = csv.reader(open('data_testing.csv'))
lines = list(reader)

#Having IP Address
print(colored('\n USING IP ADDRESS', 'grey',
attrs=['bold']))
is_valid = re.match("^(([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-
4][0-9]|25[0-5])\.){3}([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-4][0-
9]|25[0-5])$", hostname)
if is_valid == None:
      lines[1][0] = '-1'
      print(colored(' [-1]', 'green'),'Legitimate')
else:
      lines[1][0] = '1'
      print(colored(' [1]', 'red'),'Phising')

#Panjang URL
print(colored(' URL LENGTH', 'grey', attrs=['bold']))
if len(url)<54:
      lines[1][1] = '-1'
      print(colored(' [-1]', 'green'),'Legitimate')

elif len(url)<75:
      lines[1][1] = '0'
      print(colored(' [0]', 'yellow'),'Suspicious')
else:
      lines[1][1] = '1'
      print(colored(' [1]', 'red'),'Phising')

#URL Shortening Services
print(colored(' URL SHORTENING SERVICES', 'grey',
attrs=['bold']))
urlss = [
      'bit.ly',
      'goo.gl',
      'tinyurl.com',
      'buff.ly',
      'adf.ly',
      'ow.ly',
      'polr.me',
      'is.gd',
      'soo.gd',
      's2r.co'
]
for tinyurl in urlss:
  find_tinyurl = hostname.find(tinyurl)
  if find_tinyurl !=-1:
      break

if find_tinyurl !=-1:
      lines[1][2] = '1'
      print(colored(' [1]', 'red'),'Phising')
se:
      lines[1][2] = '-1'
      print(colored(' [-1]', 'green'),'Legitimate')
```

```
#Simbol "@"
print(colored(' SIMBOL "@"', 'grey', attrs=['bold']))
if url.find('@')!=-1:
      lines[1][3] = '1'
      print(colored(' [1]', 'red'),'Phising')
else:
      lines[1][3] = '-1'
      print(colored(' [-1]', 'green'),'Legitimate')

#Simbol "//"
print(colored(' SIMBOL "//"', 'grey', attrs=['bold']))
if url.find('//')>7:
      lines[1][4] = '1'
      print(colored(' [1]', 'red'),'Phising')
else:
      lines[1][4] = '-1'
      print(colored(' [-1]', 'green'),'Legitimate')

#Simbol "-"
print(colored(' SIMBOL "-"', 'grey', attrs=['bold']))

if hostname.find('-')!=-1:
      lines[1][5] = '1'
      print(colored(' [1]', 'red'),'Phising')
else:
      lines[1][5] = '-1'
      print(colored(' [-1]', 'green'),'Legitimate')

#Sub Domain
print(colored(' SUB DOMAIN', 'grey', attrs=['bold']))


negara_TLD = [
                  '.ac.',
                  '.co.',
                  '.desa.',
                  '.or.',
                  '.net.',
                  '.web.',
                  '.sch.',
                  '.go.'
]
for tld in negara_TLD:
      find_tld = hostname.find(tld)
      if find_tld !=-1:
            break

if hostname.count('.')==1:
      lines[1][6] = '-1'
      print(colored(' [-1]', 'green'),'Legitimate [a]')
elif find_tld !=-1:
      titik = hostname.count('.')-1
      if titik==1:
            lines[1][6] = '-1'
            print(colored(' [-1]', 'green'),'Legitimate
]')
```

```
      elif titik ==2:
            lines[1][6] = '0'
            print(colored(' [0]', 'yellow'),'Suspicious
[a]')
      else:
            lines[1][6] = '1'
            print(colored(' [1]', 'red'),'Phising [a]')
elif hostname.count('.')==2:
      lines[1][6] = '0'
      print(colored(' [0]', 'yellow'),'Suspicious [b]')
else:
      lines[1][6] = '1'
      print(colored(' [1]', 'red'),'Phising [b]')

#Domain Registration Length
print(colored(' DOMAIN REGISTRATION LENGTH', 'grey',
attrs=['bold']))
try:
      cd = pywhois.creation_date
      ud = pywhois.updated_date
      ed = pywhois.expiration_date

      try:
            if ud == None:
                  tahun1 = cd.year
                  tahun3 = ed.year
                  drl = tahun3-tahun1
            elif type(ud) == list:
                  string = str(ud)
                  d = string[19:23]
                  tahun3 = ed.year
                  drl = tahun3 - int(d)
            else:
                  tahun2 = ud.year
                  tahun3 = ed.year
                  drl = tahun3-tahun2
      except:
            s = ed[8:12]
            tahun1 = cd.year
            drl = int(s)-tahun1

      try:
            if drl <= 1:
                  lines[1][7] = '1'
                  print(colored(' [1]', 'red'),'Phising
[a]')
            else:
                  lines[1][7] = '-1'
                  print(colored(' [-1]',
'green'),'Legitimate')
      except:
            lines[1][7] = '1'
            print(colored(' [1]', 'red'),'Phising [b]')
cept:
    lines[1][7] = '1'
    print(colored(' [1]', 'red'),'Phising [c]')
```

```
#HTTPS Token
print(colored(' HTTPS TOKEN', 'grey', attrs=['bold']))
if hostname.find('https')!=-1:
      lines[1][8] = '1'
      print(colored(' [1]', 'red'),'Phising')
else:
      lines[1][8] = '-1'
      print(colored(' [-1]', 'green'),'Legitimate')

#SITE, DRC, IFR
try:
      page = urllib.request.urlopen(url)

      f = open('page_source.txt', 'wb')
      shutil.copyfileobj(page, f)

      print(colored(' SUBMITTING INFORMATION TO EMAIL',
'grey', attrs=['bold']))
      with open('page_source.txt', 'r') as site:
            if 'mail()' in site.read():
                  lines[1][9] = '1'
                  print(colored(' [1]', 'red'),'Phising
[a]')
            elif 'mailto:' in site.read():
                  lines[1][9] = '1'
                  print(colored(' [1]', 'red'),'Phising
[b]')
            else:
                  lines[1][9] = '-1'
                  print(colored(' [-1]',
'green'),'Legitimate')

      print(colored(' DISABLE RIGHT CLICK', 'grey',
attrs=['bold']))
      with open('page_source.txt', 'r') as drc:
            if 'contextmenu' in drc.read():
                  lines[1][10] = '1'
                  print(colored(' [1]', 'red'),'Phising
[a]')
            else:
                  lines[1][10] = '-1'
                  print(colored(' [-1]',
'green'),'Legitimate')

      print(colored(' IFRAME REDIRECTION', 'grey',
attrs=['bold']))
      with open('page_source.txt', 'r') as ifr:
            if '<iframe' in ifr.read():
                  lines[1][11] = '1'
                  print(colored(' [1]', 'red'),'Phising
]')
            else:
                  lines[1][11] = '-1'
                  print(colored(' [-1]',
reen'),'Legitimate')
```

```
except:
      print(colored(' SUBMITTING INFORMATION TO EMAIL',
'grey', attrs=['bold']))
      lines[1][9] = '1'
      print(colored(' [1]', 'red'),'Phising [c]')
      print(colored(' DISABLE RIGHT CLICK', 'grey',
attrs=['bold']))
      lines[1][10] = '1'
      print(colored(' [1]', 'red'),'Phising [b]')
      print(colored(' IFRAME REDIRECTION', 'grey',
attrs=['bold']))
      lines[1][11] = '1'
      print(colored(' [1]', 'red'),'Phising [b]')

#Age of Domain
print(colored(' AGE OF DOMAIN', 'grey', attrs=['bold']))
dn = datetime.now()
try:
      cd = pywhois.creation_date
      try:
            tahun1 = dn.year
            tahun2 = cd.year
            bulan1 = dn.month
            bulan2 = cd.month
            tahunaod = (tahun1-tahun2)*12
            bulanaod = bulan1-bulan2
            aod = (tahunaod-bulanaod)/12
      except:
            next

      try:
            if aod >= 6:
                  lines[1][12] = '-1'
                  print(colored(' [-1]',
'green'),'Legitimate')
            else:
                  lines[1][12] = '1'
                  print(colored(' [1]', 'red'),'Phising
[a]')
      except:
            lines[1][12] = '1'
            print(colored(' [1]', 'red'),'Phising [b]')
except:
      lines[1][12] = '1'
      print(colored(' [1]', 'red'),'Phising [c]')

#DNS Records
print(colored(' DNS RECORDS', 'grey', attrs=['bold']))
def get_records():
    ids = [
        'CNAME',
        'MX',
        'NS',
        'PTR',
        'SRV',
        'SOA',
```

```
        'TXT',
    ]
    for a in ids:
      try:
            answers = dns.resolver.query(hostname, a)
            return answers
      except:
            pass

dns_record = get_records()

if dns_record is None:
      lines[1][13] = '-1'
      print(colored(' [-1]', 'red'),'Phising')
else:
      lines[1][13] = '1'
      print(colored(' [1]', 'green'),'Legitimate')

#Website Traffic
xml =
urllib.request.urlopen('http://data.alexa.com/data?cli=10&d
at=s&url={}'.format(url)).read()

result= xmltodict.parse(xml)

data = json.dumps(result).replace("@","")
data_tojson = json.loads(data)

print(colored(' WEBSITE TRAFFIC', 'grey', attrs=['bold']))
try:
      url =
data_tojson["ALEXA"]["SD"][1]["POPULARITY"]["URL"]
      rank=
data_tojson["ALEXA"]["SD"][1]["POPULARITY"]["TEXT"]

      if int(rank) < 100000:
            lines[1][14] = '0'
            print(colored(' [0]', 'green'),'Legitimate')
      else:
            lines[1][14] = '-1'
            print(colored(' [-1]', 'yellow'),'Suspicious')
except:
      lines[1][14] = '1'
      print(colored(' [1]', 'red'),'Phising')

#Statistical-Reports Based Features
print(colored(' STATISTICAL-REPORTS BASED FEATURES',
'grey', attrs=['bold']))
ipaddrs = socket.gethostbyname(hostname)

p_domains = [
      'esy.es',
      'hol.es',
      '000webhostapp.com',
      '16mb.com',
      'bit.ly',
```

```
      'for-our.info',
      'beget.tech',
      'blogspot.com',
      'weebly.com',
      'raymannag.ch',
]
for domain in top_domains:
  find_domain = hostname.find(domain)
  if find_domain !=-1:
      break

with open('top_ips.csv', 'r') as top_ips:
  if str(ipaddrs) in top_ips.read():
      lines[1][15] = '1'
      print(colored(' [1]', 'red'),'Phising [a]')
  elif find_domain !=-1:
      lines[1][15] = '1'
      print(colored(' [1]', 'red'),'Phising [b]')
  else:
      lines[1][15] = '-1'
      print(colored(' [-1]', 'green'),'Legitimate')

with open('data_testing.csv', 'w') as f:
      writer = csv.writer(f)
      writer.writerows(lines)
f.close()

import numpy as np
import pandas as pd
from sklearn.svm import SVC
import pickle

# Load model
if kernel == 1:
      loaded_model = pickle.load(open('model_l.sav', 'rb'))
elif kernel == 2:
      loaded_model = pickle.load(open('model_p.sav', 'rb'))
elif kernel == 3:
      loaded_model = pickle.load(open('model_r.sav', 'rb'))
else:
      loaded_model = pickle.load(open('model_p.sav', 'rb'))

x_predict = pd.read_csv('data_testing.csv')
b = np.array(x_predict)
y_predict = b[0:,:16]

prediksi = loaded_model.predict(y_predict)
score = loaded_model.decision_function(y_predict)
non_phish = ((1-score)/2)*100
phish = (100-non_phish)

int(colored('\n RESULT:', 'grey', attrs=['bold']))
int(colored(' ======================================',
rey', attrs=['bold']))
int(colored(' PHISHING PERCENTAGE : %.2f' % phish,
rey',attrs=['bold']),colored('%','grey', attrs=['bold']))
```

```
if prediksi == 1:
      print('',colored(prediksi,'red',
attrs=['bold']),colored(' PHISHING', 'red',
attrs=['bold']))
else:
      print('',colored(prediksi, 'green',
attrs=['bold']),colored(' NON PHISHING', 'green',
attrs=['bold']))

t2 = datetime.now()
total =  t2 - t1
print(colored('\n Scanning Completed in: ', 'grey'),
colored(total, 'grey'))
```

KEMENTERIAN RISET, TEKNOLOGI DAN PENDIDIKAN TINGGI
UNIVERSITAS HASANUDDIN
KAMPUS TAMALANREA
JALAN PERINTIS KEMERDEKAAN KM. 10 MAKASSAR 90245
TELEPON (0411) 586200, 584002 FAX. (0411)

SURAT PERSETUJUAN

Nomor : 33068 /UN4.1.1.2.1.1/PK.02.03/2019

Berdasarkan Peraturan Rektor Universitas Hasanuddin tentang Penyelenggaraan Program Sarjana Nomor : 2781/UN4.1/KEP/2018 TANGGAL 16 Juli 2018, dengan ini menerangkan bahwa :

N a m a	:	DIKI WAHYUDI
Tempat/Tanggal Lahir	:	LAKALUKKU / 5 MEI 1997
Stambuk	:	D42115518
Fakultas	:	TEKNIK
Program Studi	:	TEKNIK INFORMATIKA

Telah memenuhi syarat untuk Ujian Skripsi Strata I (S1). Demikian Surat Persetujuan ini dibuat untuk digunakan dalam proses pelaksanaan ujian skripsi, dengan ketentuan dapat mengikuti Wisuda **PERIODE III MARET 2020**. **Jika persyaratan kelulusan/wisuda telah dipenuhi**. Terima Kasih.

Makassar, 19 Desember 2019
a.n.Kepala Bagian Akademik
Kepala Sub Bagian Pendidikan dan Evaluasi
Universitas Hasanuddin,

Mursalim,S.Sos
Nip. 19730216199601 1 001

Quality ISO 9001

Keterangan :

Nomor Use	:	D42115518
Nomor Password/Pin	:	32057945
Alamt Websit	:	**htt://unhas.ac.id/akad/wisuda/**
Layanan E-Mail	:	**alimkomath@gmail.com**
Catatan	:	1. Bagi Mahasiswa yang telah melaksanakan Ujian Sarjana dan dinyatakan lulus, segera menyerahkan lembar pengesahan Skripsi dan Berita Acara Ujian Sarjana ke Sub Bagian Akademik Fakultas, untuk memperoleh nomor Alumni dan didaftar sebagai Wisudawan pada periode berjalan.
		2. Jika terjadi perubahan Judul Skripsi agar melaporkan ke kasubag. Pendidikan Fakultas sebelum didaftar sebagai Wisudawan pada Periode berjalan
		3. Pada saat ON - LINE Mahasiswa diharapkan mengisi Identitas diri sesuai Surat Izin Ujian ini.
		4. Surat izin ujian hanya berlaku untuk Wisuda periode berjalan (Wisuda Maret 2020)

KEMENTERIAN RISET, TEKNOLOGI DAN PENDIDIKAN TINGGI
UNIVERSITAS HASANUDDIN
FAKULTAS TEKNIK
Kampus FakultasTeknikUnhas, Jl. PorosMalino, Gowa
http://eng.unhas.ac.id, Email : teknik@unhas.ac.id

SURAT PENUGASAN
No. 3940/UN4.7.1/DA.08.04/2019

Dari : Dekan Fakultas Teknik Universitas Hasanuddin.

Kepada. : Mereka yang tercantum namanya di bawah ini.

Isi : 1. Bahwa berdasarkan peraturan Akademik Universitas Hasanuddin Tahun 2003 Pasal 36 butir 3 point a, b (SK. Rektor Unhas Nomor : 1067 /J04/PP.08/2008), dengan ini menugaskan Saudara sebagai PANITIA SEMINAR PROPOSAL Strata Satu (S1) Departemen Teknik Informatika Fakultas Teknik Universitas Hasanuddin dengan susunan sebagai berikut :

Pembimbing I / Ketua	: 1. Dr.Eng. Muhammad Niswar, ST., M.IT
Pembimbing II / Sekretaris	: 2. A. Ais Prayogi Alimuddin, ST., M.Eng
Anggota	: 3. Dr. Ir. Zahir Zainuddin, M.Sc
	4. Adnan, ST., M.T., Ph.D

untuk menguji bagi mahasiswa tersebut di bawah ini :

Nama/NIM	: Diki Wahyudi	D421 15 518
Departemen	: Teknik Informatika	
Judul Thesis/Skripsi	: **" Aplikasi Pendeteksi Website Phising MenggunakaSupport Vector Machine (SVM) "**	

2. Waktu seminar ditetapkan oleh Panitia Seminar Proposal Strata Satu (S1).
3. Agar Surat penugasan ini dilaksanakan sebaik-baiknya dengan penuh rasa tanggung jawab.
4. Surat penugasan ini berlaku sejak tanggal ditetapkan sampai dengan berakhirnya Seminar tersebut dengan ketentuan bahwa segala sesuatunya akan ditinjau dan diperbaiki sebagaimana mestinya apabila dikemudian hari ternyata terdapat kekeliruan dalam keputusan ini.

Ditetapkan di Gowa,
Pada tanggal 4 Maret 2019
a.n. Dekan.
Wakil Dekan Bidang Akademik

Prof. Baharuddin Hamzah, ST., M.Arch., Ph.D
NIP. 19690308 199512 1 001

Tembusan :
nik Unhas
en Teknik Informatika FT-UH
bersangkutan

KEMENTERIAN RISET, TEKNOLOGI DAN PENDIDIKAN TINGGI
UNIVERSITAS HASANUDDIN
FAKULTAS TEKNIK
DEPARTEMEN TEKNIK INFORMATIKA
Kampus FakultasTeknikUnhas, Jl. PorosMalino, Gowa
http://eng.unhas.ac.id/informatika, Email : informatika@unhas.ac.id

BERITA ACARA UJIAN SEMINAR PROPOSAL

Pada hari ini **Rabu ,** tanggal **13 Maret 2019** Pukul **11.00 WITA-Selesai** bertempat di **Ruang Lab. AIMP Teknik Informatika**, telah dilaksanakan Ujian Seminar Proposal bagi Saudara :

Nama : Diki Wahyudi

No. Stambuk : D421 15 518

Fakultas/Departemen : Teknik/Teknik Informatika

Judul Skripsi : " **Aplikasi Pendeteksi Website Phising MenggunakaSupport Vector Machine (SVM)** "

Yang dihadiri oleh panitia Ujian Seminar Proposal sebagai berikut :

No.	N a m a	Jabatan	Tanda tangan
1.	Dr.Eng. Muhammad Niswar, ST., M.IT	Pemb I/Ketua	1.
2.	A. Ais Prayogi Alimuddin, ST., M.Eng	Pemb II/Sekretaris	2.
3.	Dr. Ir. Zahir Zainuddin, M.Sc	Anggota	3.
4.	Adnan, ST., M.T., Ph.D	Anggota	4.

Hasil keputusan panitia penilai Ujian Seminar Proposal Tugas Akhir : **Lulus** / ~~**Tidak lulus**~~ dengan nilai angka .....86..... dan huruf .....A.....

Gowa, 13 Maret 2019
Ketua/Sekretaris Panitia Ujian,

Dr.Eng. Muhammad Niswar, ST., M.IT

KEMENTERIAN RISET, TEKNOLOGI DAN PENDIDIKAN TINGGI
UNIVERSITAS HASANUDDIN
FAKULTAS TEKNIK
DEPARTEMEN TEKNIK INFORMATIKA
Kampus FakultasTeknikUnhas, Jl. PorosMalino, Gowa
http://cng.unhas.ac.id/informatika, Email : informatika@unhas.ac.id

SURAT KETERANGAN NILAI UJIAN SEMINAR PROPOSAL
Nomor : 263.../UN4.7.7.TI/DA.04.09/2019

Pada hari ini **Rabu** , tanggal **13 Maret 2019** Pukul **11.00 WITA-Selesai** bertempat di **Ruang Lab. AIMP Teknik Informatika**, telah dilaksanakan Ujian Seminar Proposal bagi Saudara :

Nama	: Diki Wahyudi
No. Stambuk	: D421 15 518
Fakultas/Departemen	: Teknik/Teknik Informatika
Judul Skripsi	: " **Aplikasi Pendeteksi Website Phising MenggunakaSupport Vector Machine (SVM)** "

Setelah pembawa ujian seminar proposal menguraikan tugas akhirnya dan menjawab pertanyaan dengan dinyatakan lulus/tidak lulus, Baik/Cukup/Sedang.

Maka berdasarkan hasil penilaian dinyatakan lulus / ~~tidak lulus~~

Dengan nilai :

A — A- — B+ — B — B- — C+ — C — D — E

Mengetahui:
A.n. Ketua Departemen Teknik Informatika
Sekteratis Departemen

Dr. Indrabayu, ST., M.T., M.Bus,Sys
Nip.197507162002121004

Dosen Penguji,

Dr.Eng. Muhammad Niswar, ST., M.IT
Nip.197309221999031001

Diketahui oleh:
Dekan
Wakil Dekan Bidang Akademik,

Prof. Baharuddin Hamzah, ST., M.Arch., Ph.D
Nip. 19690308 199512 1 001

KEMENTERIAN RISET, TEKNOLOGI DAN PENDIDIKAN TINGGI
UNIVERSITAS HASANUDDIN
FAKULTAS TEKNIK
DEPARTEMEN TEKNIK INFORMATIKA
Kampus FakultasTeknikUnhas, Jl. PorosMalino, Gowa
http://eng.unhas.ac.id/informatika, Email : informatika@unhas.ac.id

DAFTAR HADIR SEMINAR HASIL

Nama/Stambuk : 1. Diki Wahyudi D421 15518

Judul Skipsi/T.A : **"Aplikasi Pendeteksi Website Phising Menggunakan Spport Vector Machine (SVM)"**

Hari/Tanggal : Rabu, 4 Desember 2019
Jam : 09.00 Wita – Selesai
Tempat : Ruang Lab. **UBICON** Departeman Teknik Informatika Gowa

No.	Jabatan	Nama Dosen	Tanda Tangan
I.	Pembimbing I	1. Dr.Eng. Muhammad Niswar,ST.,M.I.T	1.....
	Pembimbing II	2. A. Ais Prayogi Alimuddin,ST.M.Eng	2.....
II.	Anggota Penguji	3. Dr. Ir. Zahir Zainuddin,M.Sc	3.....
		4. Adnan,ST.,M.T.Ph.D	.....

PANITIA UJIAN

Ketua,

Dr.Eng. Muhammad Niswar,ST.,M.I.T

Sekretaris,

A. Ais Prayogi Alimuddin,ST.M.Eng

KEMENTERIAN RISET, TEKNOLOGI DAN PENDIDIKAN TINGGI
UNIVERSITAS HASANUDDIN
FAKULTAS TEKNIK
DEPARTEMEN TEKNIK INFORMATIKA
Kampus FakultasTeknikUnhas, Jl. PorosMalino, Gowa
http://eng.unhas.ac.id/informatika, Email : informatika@unhas.ac.id

BERITA ACARA SEMINAR HASIL

Pada hari ini **Rabu,** tanggal **4 Desember 2019** Pukul **09.00 WITA - Selesai** bertempat di **Ruang Lab.UBICON Departemen Teknik Informatika,** telah dilaksanakan Seminar Hasil bagi Saudara :

Nama : Diki Wahyudi
No. Stambuk : D421 15518
Fakultas/Departemen : Teknik/Teknik Informatika
Judul Skripsi : **"Aplikasi Pendeteksi Website Phising Menggunakan Spport Vector Machine (SVM)"**

Yang dihadiri oleh Tim Penguji Seminar Hasil sebagai berikut :

No.	N a m a	Jabatan	Tanda tangan
1.	Dr.Eng. Muhammad Niswar,ST.,M.I.T	Pemb I/Ketua	1.
2.	A. Ais Prayogi Alimuddin,ST.M.Eng	Pemb II/Sekretaris	2.
3.	Dr. Ir. Zahir Zainuddin,M.Sc	Anggota	3.
4.	Adnan,ST.,M.T.Ph.D	Anggota	4.

Hasil keputusan Tim Penguji Seminar Hasil : **Lulus / ~~Tidak lulus~~** dengan nilai angka ...87..... dan huruf .....A.....

Makassar, 4 Desember 2019

~~Ketua~~/Sekretaris Panitia Ujian,

Dr.Eng. Muhammad Niswar,ST.,M.I.T

KEMENTERIAN RISET, TEKNOLOGI DAN PENDIDIKAN TINGGI
UNIVERSITAS HASANUDDIN
FAKULTAS TEKNIK
DEPARTEMEN TEKNIK INFORMATIKA
Kampus FakultasTeknikUnhas, Jl. PorosMalino, Gowa
http://eng.unhas.ac.id/informatika, Email : informatika@unhas.ac.id

SURAT KETERANGAN NILAI SEMINAR HASIL

Nomor : 735 / UN4.7.7.TI/PK.03.06/2019

Pada hari ini **Rabu,** tanggal 4 **Desember 2019** Pukul **09.00 WITA - Selesai** bertempat di **Ruang Lab.UBICON Departemen Teknik Informatika,** telah dilaksanakan Seminar Hasil bagi Saudara :

Nama : Diki Wahyudi
No. Stambuk : D421 15518
Fakultas/Departemen : Teknik/Teknik Informatika
Judul Skripsi : **"Aplikasi Pendeteksi Website Phising Menggunakan Spport Vector Machine (SVM)"**

Setelah pembawa seminar hasil menguraikan tugas akhirnya dan menjawab pertanyaan dari Tim Penguji dinyatakan Lulus / Tidak Lulus dengan nilai :

A ~~A-~~ ~~B+~~ ~~B~~ ~~B-~~ ~~C+~~ ~~C~~ ~~D~~ ~~E~~

Mengetahui:

Ketua Departemen Tek.Informatika,

Dr. Amil Ahmad Ilham,ST.,M.I.T
Nip. 19731010 199802 1 001

Dosen Penguji,

Dr.Eng. Muhammad Niswar,ST.,M.I.T
Nip. 19730922 199903 1 001

Diketahui oleh,
a.n Dekan,
Wakil Dekan Bidang Akademik,Riset dan Inovasi

Prof. Baharuddin Hamzah,ST., M.Arch., Ph.D
Nip. 19690308 199512 1 001

KEMENTERIAN RISET, TEKNOLOGI DAN PENDIDIKAN TINGGI
UNIVERSITAS HASANUDDIN
FAKULTAS TEKNIK
DEPARTEMEN TEKNIK INFORMATIKA
Kampus FakultasTeknikUnhas, Jl. PorosMalino, Gowa
http://eng.unhas.ac.id/informatika, Email : informatika@unhas.ac.id

DAFTAR HADIR UJIAN SKRIPSI MAHASISWA FAKULTAS TEKNIK UNHAS

Nama/Stambuk : Diki Wahyudi D421 15518

Judul Skipsi/T.A : **"Aplikasi Pendeteksi Website Phishing Menggunakan Machine Learning"**

Hari/Tanggal : Jum'at, 3 Januari, 2020

Jam : 09.00 Wita – Selesai

Tempat : Ruang Lab.UBICON Departemen Teknik Informatika

No.	Jabatan	Nama Dosen	Tanda Tangan
L.	Pembimbing I	1. Dr.Eng.Muhammad Niswar,ST.,M.I.T	1.....
	Pembimbing II	2. A.Ais Prayogi Alimuddin,ST.M.Eng	2 .....
II.	Anggota Penguji	3. Dr. Ir. Zahir Zainuddin,M.Sc	3 .....
		4. Adnan,ST.,M.T.Ph.D	4 .....

PANITIA UJIAN

Ketua,

Dr.Eng.Muhammad Niswar,ST.,M.I.T

Sekretaris,

A.Ais Prayogi Alimuddin,ST.M.Eng

KEMENTERIAN RISET, TEKNOLOGI DAN PENDIDIKAN TINGGI
UNIVERSITAS HASANUDDIN
FAKULTAS TEKNIK
DEPARTEMEN TEKNIK INFORMATIKA
Kampus FakultasTeknikUnhas, Jl. PorosMalino, Gowa
http://eng.unhas.ac.id/informatika, Email : informatika@unhas.ac.id

BERITA ACARA UJIAN SKRIPSI

Pada hari ini **Jum'at,** tanggal **3 Januari 2020** Pukul **09.00 WITA - Selesai** bertempat di **Ruang Lab.UBICON Departemen Teknik Informatika,** telah dilaksanakan Ujian Skripsi bagi Saudara :

Nama : Diki Wahyudi
No. Stambuk : D421 15518
Fakultas/Departemen : Teknik/Teknik Informatika
Judul Skripsi : **"Aplikasi Pendeteksi Website Phishing Menggunakan Machine Learning"**

Yang dihadiri oleh Tim Penguji Ujian Skripsi sebagai berikut :

No.	N a m a	Jabatan	Tanda tangan
1.	Dr.Eng.Muhammad Niswar,ST.,M.I.T	Pemb I/Ketua	1.
2.	A.Ais Prayogi Alimuddin,ST.M.Eng	Pemb II/Sekretaris	2.
3.	Dr. Ir. Zahir Zainuddin,M.Sc	Anggota	3.
4.	Adnan,ST.,M.T.Ph.D	Anggota	4.

Hasil keputusan Tim Penguji Ujian Skripsi/Tugas Akhir : **Lulus** / ~~**Tidak lulus**~~ dengan nilai angka ..... dan huruf .....

Gowa, 3 Januari, 2020

Ketua/Sekretaris Panitia Ujian,

Dr.Eng.Muhammad Niswar,ST.,M.I.T

KEMENTERIAN RISET, TEKNOLOGI DAN PENDIDIKAN TINGGI
UNIVERSITAS HASANUDDIN
FAKULTAS TEKNIK
DEPARTEMEN TEKNIK INFORMATIKA
Kampus FakultasTeknikUnhas, Jl. PorosMalino, Gowa
http://eng.unhas.ac.id/informatika, Email : informatika@unhas.ac.id

SURAT KETERANGAN NILAI UJIAN SKRIPSI
Nomor : 03 / UN4.7.7.TI/PK.03.06/2020

Pada hari ini **Jum'at,** tanggal **3 Januari, 2020** Pukul **09.00 WITA - Selesai** bertempat di **Ruang Lab.UBICON Departemen Teknik Informatika,** telah dilaksanakan Ujian Skripsi bagi Saudara :

Nama : Diki Wahyudi
No. Stambuk : D421 15518
Fakultas/Departemen : Teknik/Teknik Informatika
Judul Skripsi : **"Aplikasi Pendeteksi Website Phishing Menggunakan Machine Learning"**

Setelah pembawa ujian Skripsi menguraikan tugas akhirnya dan menjawab pertanyaan dari Tim Penguji dinyatakan Lulus / Tidak Lulus dengan nilai :

A ~~A-~~ ~~B+~~ ~~B~~ ~~B-~~ ~~C+~~ ~~C~~ ~~D~~ ~~E~~

87

Mengetahui:

Ketua Departemen Tek.Informatika,

Dr. Amil Ahmad Ilham,ST M.I.T
Nip. 19731010 199802 1 001

Dosen Penguji,

Dr.Eng.Muhammad Niswar,ST.,M.I.T
Nip. 19730922 199903 1 001

Diketahui oleh,
a.n. Dekan,
Wakil Dekan Bidang Akademik,Riset dan Inovasi

Prof. Baharuddin Hamzah,ST., M.Arch., Ph.D
Nip. 19690308 199512 1 001

KARTU BIMBINGAN SKRIPSI

Prodi S1 Teknik Informatika Universitas Hasanuddin

Stb.	Nama Mahasiswa
D42115518	Diki Wahyudi

Pembimbing.	Nama Pembimbing	Paraf & Tgl. Persetujuan Ujian Akhir
I	Dr. Eng. Muhammad Niswar, ST., M.IT	
II	A. Ais Prayogi Alimuddin, ST., M.Eng	
No SK Pemb:		

Judul Skripsi:	Aplikasi Pendeteksi Website Phishing Menggunakan Machine Learning

No.	Tanggal Bimbingan	Uraian Kegiatan Bimbingan	Paraf Pemb
1	15 / 09 / 2019	Konsultasi Dataset	
2	24 / 09 / 2019	Konsultasi metode ekstraksi fitur	
3	07 / 05 / 2019	Konsultasi metode yang akan digunakan	
4	22 / 09 / 2019	Konsultasi hasil pengujian sistem	
5	06 / 10 / 2019	Bimbingan Penulisan skripsi	
6	12 / 11 / 2019	Bimbingan pengujian sistem	
7	25 / 11 / 2019	Bimbingan skripsi	
8	26 / 11 / 2019	Bimbingan skripsi	
9	04 / 12 / 2019	Diskusi revisi skripsi	
	/ 2019	Asistensi hasil revisi seminar hasil	
	/ 2019	Asistensi hasil revisi seminar hasil	

LEMBAR PERBAIKAN SKRIPSI

"APLIKASI PENDETEKSI WEBSITE PHISHING MENGGUNAKAN MACHINE LEARNING"

OLEH:

DIKI WAHYUDI
NIM D42115518

Skripsi ini telah dipertahankan pada Ujian Akhir Sarjana tanggal 04 Januari 2020.

Telah dilakukan perbaikan penulisan dan isi skripsi berdasarkan usulan dari penguji dan pembimbing skripsi.

Persetujuan perbaikan oleh tim penguji:

	Nama	Tanda Tangan
Ketua	Dr. Eng. Muhammad Niswar, ST., M.IT.	
Sekretaris	A. Ais Prayogi Alimuddin, ST., M.Eng.	
Anggota	Dr. Ir. Zahir Zainuddin, M.Sc.	
	Adnan, ST., M.T., Ph.D.	

Persetujuan Perbaikan oleh pembimbing:

Pembimbing	Nama	Tanda Tangan
I	Dr. Eng. Muhammad Niswar, ST., M.IT.	
II	A. Ais Prayogi Alimuddin, ST., M.Eng.	