Python visualisointityökaluna

Ammattikorkeakoulututkinnon opinnäytetyö

Tietojenkäsittelyn koulutus

Syksy 2024

Rasmus Peltoranta

HAMK
Hämeen ammatti-
korkeakoulu

Tiivistelmä

Tietojenkäsittelyn koulutus
Tekijä Rasmus Peltoranta Vuosi 2024
Työn nimi Python visualisointityökaluna
Ohjaaja Tommi Lahti

Tämän opinnäytetyön tarkoituksena oli tutkia datan visualisointia ja sen hyödyntämistä interaktiivisten sovellusten kehittämisessä. Tutkimuksessa keskityttiin datan visualisointiin, sen eri vaiheisiin ja työkaluihin. Opinnäytetyöllä ei ole toimeksiantajaa.

Opinnäytetyön ensimmäisessä teoreettisessa osassa käydään läpi millaista dataa on olemassa, sen peruskäsitteitä ja käsittelyä, datan visualisointia ja yleisimpiä kaaviomalleja sekä datavastuullisuutta ja harhaanjohtavaa visualisointia. Toisessa teoreettisessa osassa käydään läpi Python-ohjelmointikieltä datan visualisointityökaluja ja sen eri kirjastoja Matplotlib, Seaborn ja isoimmassa osassa opinnäytetyössä oleva Plotly.

Opinnäytetyön tyyppi on toiminnallinen, sillä käytännön osassa tehtiin Plotlyn Dash-kehyksellä sovellus, jonka avulla voidaan kätevästi tarkastella Englannin Valioliigan pelaajien tilastoja.

Tutkimuksen tulokset osoittivat, että oikeanlaisten visualisointityökalujen valinta ja niiden tehokas käyttö voivat merkittävästi parantaa datan ymmärtävyyttä ja analysoitavuutta. Johtopäätöksenä voidaan todeta, että Plotly on erityisen hyödyllinen interaktiivisten ja dynaamisten visualisointien luomisessa.

Avainsanat Visualisointi, Python, Plotly
Sivut 26 sivua ja liitteitä 1 sivu

HAMK
Hämeen ammatti-
korkeakoulu

Abstract

DP	Degree Programme in Business Information Technology	
Author	Rasmus Peltoranta	Year 2024
Subject	Python as a Visualization Tool	
Supervisor	Tommi Lahti	

The purpose of this thesis was to study data visualization and its utilization in the development of interactive applications. The research focused on data visualization, its various stages, and tools. The thesis does not have a commissioning party; it is based on the author's own subject of interest.

In the first theoretical part of the thesis, the types of data, its basic concepts, processing, data visualization, common chart models, as well as data responsibility and misleading visualizations are discussed. The second theoretical part covers the Python programming language, data visualization tools, and various libraries such as Matplotlib, Seaborn, and Plotly, which plays the largest role in this thesis.

The thesis is of a functional type, as the practical part involved creating an application using t he Plotly Dash framework. This application allows for convenient examination of the statistics of players in the English Premier League.

The research results demonstrated that the selection of appropriate visualization tools and their effective use can significantly improve the comprehensibility and analyzability of data. As a conclusion, it can be stated that Plotly is particularly useful for creating interactive and dynamic visualizations.

Keywords	Visualization, Python, Plotly
Pages	26 pages and appendices 1 page

Sanasto

Visualisointi: Datan esittäminen graafisessa muodossa, kuten kaavioina, jotta monimutkainen tieto on helpommin ymmärrettävää.

Python: Yksinkertainen ja suosittu ohjelmointikieli, jota käytetään laajasti mm. data-analytiikassa ja visualisoinnissa.

Plotly: Pythonin kirjasto, jonka avulla voidaan luoda sekä staattisia että interaktiivisia visualisointeja. Erityisen hyödyllinen dynaamisissa sovelluksissa.

Matplotlib: Python-kirjasto kaavioiden ja datavisualisointien luomiseen. Yksi vanhimmista ja eniten käytetyistä työkaluista tieteellisessä laskennassa.

Seaborn: Python-kirjasto, joka rakentuu Matplotlibin päälle ja tarjoaa yksinkertaisen tavan luoda tilastollisia visualisointeja, integroituna Pandas DataFrameen.

DataFrame: Pandas-kirjaston tietorakenne, joka mahdollistaa taulukkomuotoisen datan helpon käsittelyn ja analysoinnin.

Dash: Plotlyn kehittämä avoimen lähdekoodin web-kehys, jonka avulla voi rakentaa interaktiivisia datavisualisointisovelluksia Pythonilla.

Bootstrap: Visuaalisen tyylityksen kehys, jota käytetään web-sovellusten ulkoasun muokkaamiseen. Dash käyttää Bootstrapia tuodakseen visuaalista ilmettä sovelluksiin.

CSV (Comma-Separated Values): Tiedostomuoto, jossa data tallennetaan tekstitiedostoon erottamalla kentät toisistaan pilkuilla. Käytetään yleisesti taulukkomuotoisen datan siirtämiseen ja säilyttämiseen.

ETL (Extract, Transform, Load): Prosessi, jossa data erotetaan eri lähteistä, muunnetaan yhteensopivaan muotoon ja ladataan analysoitavaksi.

GDPR (General Data Protection Regulation): Euroopan unionin tietosuoja-asetus, joka säätelee henkilötietojen keräämistä ja käyttöä.

Datan esikäsittely: Vaiheet, kuten puhdistaminen ja integrointi, joita suoritetaan ennen varsinaista datan analysointia, jotta raakadatasta saadaan hyödyllistä ja luotettavaa tietoa.

Hajontakaavio: Kaaviotyyppi, jossa kaksi muuttujaa esitetään pisteinä xy-koordinaatistossa, yleensä havainnollistamaan näiden välistä suhdetta.

Pylväskaavio: Kaavio, joka esittää dataa pystysuorilla tai vaakasuorilla suorakaiteilla, joissa kunkin pylvään korkeus tai leveys kuvastaa mitattavaa arvoa.

Histogrammi: Kaavio, joka näyttää jatkuvaa dataa jakautuneena intervalleihin suorakulmaisina sarakkeina, usein käytetty datan jakauman arvioimiseen.

Sisällys

Kuvat

Liitteet

1 Johdanto

Tämän opinnäytetyön tavoitteena on tutkia datan visualisointia ja sen hyödyntämistä interaktiivisten sovellusten kehittämisessä. Datan visualisointi auttaa hahmottamaan suuria tietomääriä, ja oikeiden kaaviomallien avulla voidaan esittää selkeästi tilastollisia havaintoja. Työn teoriaosassa käsitellään datan visualisoinnin perusteita sekä erilaisia kaaviomalleja, kuten pylväs- ja pistekaavioita, ja niiden soveltuvuutta eri tilanteisiin.

Lisäksi esitellään kolme Python-kirjastoa: Matplotlib, Seaborn ja Plotly. Matplotlib ja Seaborn ovat erinomaisia staattisten visualisointien luomiseen, kun taas Plotly erottuu interaktiivisilla ja dynaamisilla kaavioillaan, jotka soveltuvat erityisesti verkkosovelluksiin.

Opinnäytetyön käytännön osassa rakennetaan Plotlyn Dash-kehystä hyödyntäen interaktiivinen sovellus Valioliigan pelaajatilastojen tarkasteluun ja vertailuun. Sovelluksen avulla käyttäjät voivat analysoida eri tilastoja, kuten maaleja, syöttöjä ja torjuntoja, sekä vertailla pelaajia toisiinsa. Sovellus tarjoaa käyttäjäystävällisen ja joustavan tavan visualisoida dataa ja tehdä siitä helposti ymmärrettävää.

Tutkimuskysymykset:

- Miksi datan visualisointi on tärkeää?
- Millä tavoin dataa voidaan visualisoida?
- Mitä vaiheita datan visualisoimiseen liittyy?

2 Datan visualisointi. Perusteet ja merkitys

Datan visualisointi muodostaa olennaisen osan tiedon analysointia ja viestintää. Sen avulla monimutkainen tietomäärä voidaan selkeästi esittää graafisesti, mikä tehostaa käyttäjien kykyä hahmottaa ja tulkita tietoa tehokkaasti. Visualisointityökalut ja kirjastot tarjoavat erilaisia mahdollisuuksia datan esittämiseen, ja niiden valinta riippuu usein itse datasta sekä siitä, millaisia tavoitteita halutaan saavuttaa visualisoinnin avulla.

2.1 Datan peruskäsitteet

Dataa kerätään suuria määriä joka päivä ja määrät kasvavat mitä pidemmälle ajassa menemme. 90% datasta on kerätty viimeisen kahden vuoden aikana. Dataa on paljon erilaista ja eri tarkoituksiin. Seuraavaksi tutustutaan millaista dataa on ja mitä ne tarkoittavat. (Vij, 2022)

Laadullinen data (eng. ”Qualitive”), jota kutsutaan myös kategorialliseksi dataksi, on tietoa, joka ei ole mitattavissa numeroilla. Laadullista dataa voi lajitella ainoastaan kategorioihin, kuten esimerkiksi sukupuoli, siviilisääty tai koulutustaso.

Määrällinen data (eng. ”Quantitative”) on mitattavissa olevaa numeerista dataa. Määrällistä dataa voidaan lajitella suurimmasta pienempään, kuvata graafisesti tai käyttää matemaattisissa analyyseissä. Määrällisen datan tyypillisiä esimerkkejä ovat aika, paino, pituus ja lämpötila. (Vij, 2022)

Laadullista dataa on kahta erilaista tyyppiä. Nominaalidata on datatyyppi, jossa jokainen muuttuja on itsenäinen ja niitä ei voi verrata toisiinsa. Nominaalidataa on esimerkiksi silmien väri. Vaikka silmien värejä on olemassa monia, emme voi kuitenkaan sanoa, että ne ovat erilaisia suhteessa toisiinsa.

Järjestysdata on laadullisen datan tyyppi, jossa muuttujat ovat luonnollisessa suhteessa toistensa kanssa. Muuttujat ovat suhteellisen erilaisia toisiinsa nähden, on kyse sitten mistä tahansa mittayksiköstä. Esimerkiksi koulutustaso on järjestysdataa. Maisterin, kandidaatin ja tohtorin tutkinnot ovat eri mittaisia eli vaativat eri määrän aikaa, joten niillä voisi suorittaa matemaattisia operaatiota. Edelliseen esimerkkiin viitaten järjestysdata saattaa joskus olla jotakin laadullisen ja määrällisen datan väliltä.

Määrällistä dataa on myös kahdenlaista. Diskreetti data on dataa, joka koostuu toisistaan erillään olevista kokonaisluvuista. Diskreettejä arvoja voidaan laskea, sillä niitä on tarkka

joukko, mutta ne eivät ole mitattavissa. Esimerkiksi ei voida aina jakaa luokassa olevien ihmisten määrää kahtia, sillä ihmistä et voi puolittaa.

Jatkuva dataa on esimerkiksi aika, korkeus tai hinta. Jokainen arvo pystytään jakamaan tai pienentämään ja se pysyy edelleen pätevänä. Se, kuinka korkea joku on tai kuinka kauan hänellä menee jonkin asian suorittamiseen, voidaan jakaa millimetreihin tai sekunnin tuhannesosiin saakka. (Vij, 2022)

2.2 Datasta tiedoksi

Tiedonlouhinta tai datan kaivaminen tarkoittaa tiedon eristämistä tai hakemista suurista datamääristä. Jotta suurista määristä raakaa dataa saadaan ymmärrettävää tietoa, se käy läpi seitsemän erilaista vaihetta. Nämä vaiheet ovat datan puhdistaminen, datan integrointi, datan valinta, datan muuntaminen, datan kaivaminen, kaavion arviointi ja tiedon esittäminen. Ensimmäiset neljä vaihetta on datan esikäsittelyn vaiheita, ennen kuin dataa aletaan kaivamaan. (Han & Kamber, 2006)

2.2.1 Datan esikäsittely

Monet yritykset hankkivat suuria määriä dataa itselleen omiin ”datajärviinsä” tai datavarastoihinsa tavoitteenaan mahdollistaa parempia analytiikkaprosessaja. Datan keruuseen kuitenkin sisältyy monia erilaisia haasteita, kuten kirjoitusvirheet, puuttuvat arvot, kaksoiskappaleet jne. Apulaisprofessori Xu Chun (2019) mukaan vuonna 2017 Kagglen tuottama kysely datatieteen ja koneoppimisen tilasta paljastaa, että datan epäpuhtaus on yleisin haaste, jonka parissa työntekijät kamppailevat datankäsittelyssä. (Ilyas & Chu, 2019)

Datan integrointi tarkoittaa datan yhdistämistä eri lähteistä samaan paikkaan, kuten datavarastoon, josta sitä voidaan käyttää liiketoiminnan tarpeisiin. (Underdahl, 2018)

Datan integrointiin liittyy vahvasti ETL-prosessi eli ”Extract, Transform, Load”, suomeksi ”Erota, Muunna, Lataa”. Hyvä ETL-työkalu voi erottaa tiedon useista tietolähteistä, muuntaa sen yhtenäiseen muotoon ja ladata kohdejärjestelmään, kuten pilvipohjaiseen tietovarastoon tai datajärveen. (Hagberg, 2023)

2.3 Datan visualisointi

Ihmiskunta on pyrkinyt visualisoimaan ympärillään tapahtuvia asioita jo tuhansien vuosien ajan jo kauan ennen kirjoitustaitoa, silmämme eivät pysty suoraan näkemään kehon toimintoja, sosiaalisia tai ekonomisia rakenteita mutta, kun ne esitetään visuaalisessa muodossa, niistä tulee helpompia ymmärtää(Koponen & Hilden, 2019). Koposen ja Hildenin mukaan on laskettu, että visuaaliset aistimme lähettävät n. 8 kertaa enemmän tietoa aivoillemme kuin muut aistit yhteensä.

Visualisointi perustuu dataan, joka ei ole näkyvissä normaaleissa tilanteissa, kuten tilastot. Visualisointiprosessin ensisijainen tehtävä on synnyttää kuva. Kuitenkaan kuva, joka esittää suoraa havaintoa, kuten maisema ei ole visualisointia. Kuvan tulisi olla tunnistettavissa informaatioksi ja informaatio tulisi esittää mahdollisimman yksiselitteisesti (Koponen & Hilden 2019).

Koponen ja Hilden jakavat visuaaliset esitykset kahteen kategoriaan: selittävään ja tutkivaan. Selittävän esityksen päätarkoitus on jakaa tietoa ihmisten kesken. Niitä käytetään osoittamaan, selittämään ja tuomaan esille faktoja. Esityksen tekijä on tietoinen faktoista ja pyrkii esittämään ne mahdollisimman selkeästi yleisölle.

Tutkivan esityksen tarkoitus on tuoda esille havaintoja ja analyysejä tiedosta. Tutkivan esityksen tehtävä ei ole tuoda esille viestiä, jonka esityksen tekijä on laatinut etukäteen, vaan niiden tarkoitus on toimia työkaluna, josta lukija voi itse löytää ja poimia mielenkiintoista dataa.

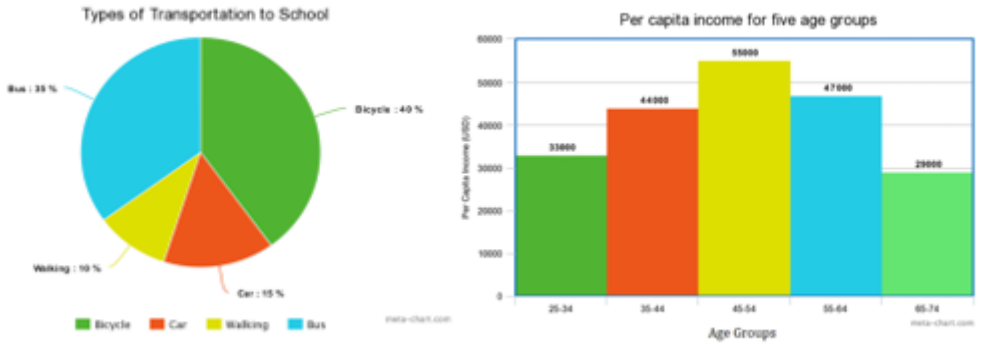
2.4 Tilastollisten kaavioiden yleiset tyypit

Kaavioita datan visualisointiin on todella paljon ja erilaisia. Tässä kappaleessa käymme läpi 5 yleisintä kaaviotyyppiä. Ensimmäisenä yksi yleisimmistä kaaviotyypeistä on kuvan 1 viivakaavio. Viivakaavio koostuu pisteistä, jotka ovat yhdistetty viivoilla. Sitä käytetään kuvaamaan esimerkiksi muutosta vuosien varrella, kuten esimerkiksi myyntiä tai asuntojen hintoja yms. Toinen hyvin käytetty kaaviomuoto on kuvan 1 pylväskaavio. Pylväskaavion on tarkoitus esittää diskreettiä dataa suorakaiteen muotoisilla sarakkeilla. Pylväskaavioita näkee usein taloustieteessä, tilastotieteessä ja markkinoinnissa. Pylvään korkeus vastaa niiden edustamia arvoja, kuten esimerkiksi myyntiä. Pylväskaavion vaakasuora akseli esittää diskreettejä kategorioita ja pystyakseli näyttää niiden mitatun arvon. (Valcheva, 2017)

Kuva 1. Viivakaavio ja pylväskaavio

Kuvassa 2 oleva piirakkakaavio esittää datan lohkojen muodossa ja havainnollistaa niiden suhdetta toisiin lohkoihin. Kaavio kokonaisuudessaan on 100% ja yksi lohko on jokin tietty osuus siitä. Esimerkkinä piirakkakaavio voi kuvata eri kuljetustapojen osuutta, joita opiskelijat käyttävät päästäkseen kouluun. Histogrammi esittää jatkuvaa dataa järjestettyinä suorakulmaisina sarakkeina, kuten kuvassa 2. Yleensä histogrammissa ei ole aukkoja sarakkeiden välissä. Histogrammit näyttävät vähän pylväskaavioita. Niillä on kuitenkin selkeä ero, pylväskaaviot edustavat kategorista dataa ja histogrammit jatkuvaa dataa. (Valcheva, 2017)

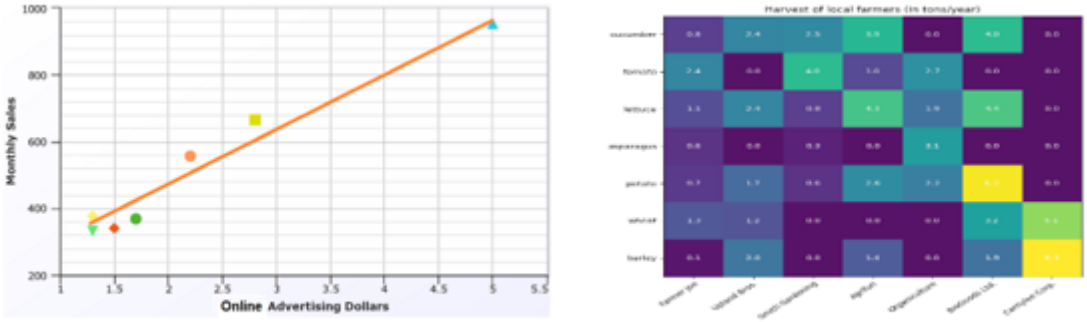
Kuva 2. Piirakkakaavio ja histogrammi.



Hajontakaavio on x-y-diagrammi, joka näyttää kahden muuttujan välisen suhteen, kuten näemme kuvassa 3. Sitä käytetään kuvaamaan datapisteitä sekä vaakasuoralla x-akselilla että pystysuoralla y-akselilla. Hyvin usein mutta ei aina, ensimmäistä muuttuja kutsutaan riippumattomaksi muuttujaksi ja toista riippuvaiseksi, sillä sen arvot riippuvat ensimmäisestä muuttujasta. Tämän kaltaisissa tapauksissa hajontakaavio mahdollistaa muuttujien välisen riippuvuuden visualisoinnin. (Valcheva, 2017)

Lämpökartta on kaksiulotteinen datan visualisointimenetelmä, jossa numeeriset arvot esitetään värien avulla. Värit voivat vaihdella joko saman sävyn eri intensiteeteissä tai käyttää erilaisia värejä väripaletista. Lämpökartoilla on useita etuja. Ne ovat silmiinpistäviä ja herättävät mielenkiintoa värienkäytöllään. Lisäksi ne mahdollistavat datan tarkastelun yksityiskohtaisemmin kuin perinteiset viiva- tai pylväskaaviot. Huolimatta yksityiskohtaisuudestaan lämpökartat ovat helposti ymmärrettäviä ja tarjoavat yleiskuvan datasta ilman tarkkoja numeroarvoja. Lämpökartoilla on kuitenkin yksi sisäsyntyinen heikkous: tarkkoja numeroarvoja on vaikea erottaa, vaikka käytössä olisikin jatkuva asteikko. Tämä johtuu siitä, että ihmisen näköhavainto ei pysty täysin tarkkaan arvioimaan eri värisävyjen intensiteettejä. Silti lämpökartat ovat hyödyllinen työkalu datavisualisoinnissa, kunhan niitä käytetään oikein. (Sukumar, 2024)

Kuva 3. Hajontakaavio ja värikartta.



2.5 Datavastuullisuus

Ihmislähtöinen ja reilu datatalous pyrkii antamaan yksilöille valtaa omien tietojensa käyttöön osana oikeudenmukaista digitaalista ympäristöä. Yksilöt nähdään aktiivisina toimijoina, ei vain tietolähteinä. Yritysten on toimittava arvostavasti, reilusti ja läpinäkyvästi niin asiakkaitaan kuin kumppaneitaankin kohtaan myös datan käsittelyssä. Tulevaisuudessa Euroopassa dataa jaetaan vapaammin menestyvissä ekosysteemeissä, mutta reilut datakumppanuudet perustuvat lupauksiin, saumattomuuteen ja avoimuuteen yhteisten sopimusten perusteella.

Yrityksille data on tärkeä resurssi, jonka vastuullinen ja älykäs käyttö tuo etuja sekä yritykselle että sen sidosryhmille. Datavastuullisuus katsotaan leikkaavana toimintana yritysvastuun kaikilla osa-alueilla. Yritysvastuu ulottuu niin taloudelliseen, sosiaaliseen kuin ympäristövastuuseenkin, ja se vaatii toimintaa lain, asetusten ja sääntöjen mukaisesti sekä läpinäkyvää raportointia. (Parikka & Härkönen, 2020)

GDPR eli yleinen tietosuoja-asetus sisältää seitsemän periaatetta, jotka ohjaavat henkilötietojen käsittelyä. Nämä periaatteet ovat keskeisiä yrityksille, jotka käsittelevät henkilötietoja, ja niiden noudattaminen on välttämätöntä lain mukaisuuden varmistamiseksi. Näitä periaatteita ovat laillisuus, oikeudenmukaisuus ja läpinäkyvyys, tarkoitussidonnaisuus, tietojen minimointi; tarkkuus, säilytyksen rajoittaminen, eheys ja luottamuksellisuus sekä vastuullisuus.

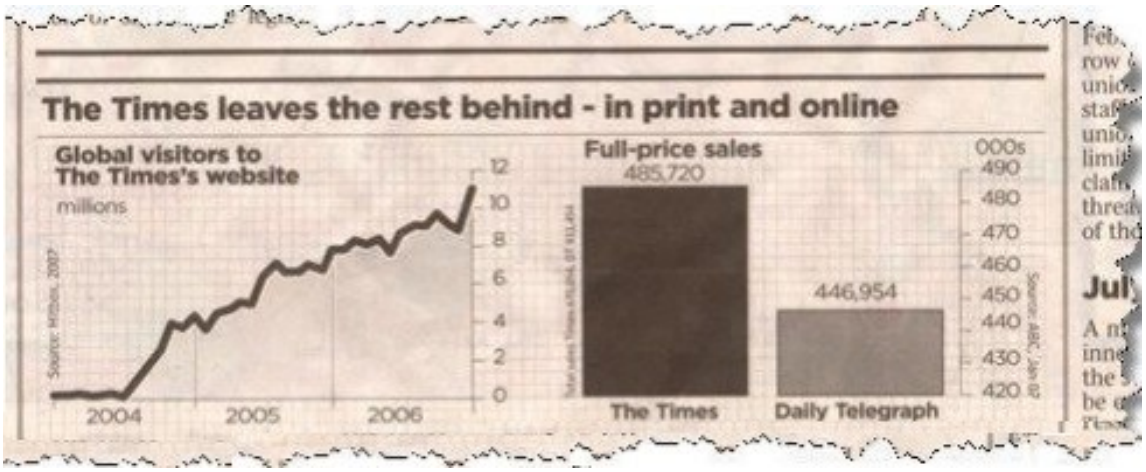
Esimerkiksi, kun yritys kerää henkilötietoja uutiskirjeen tilaajilta verkkosivustonsa kautta, sen on varmistettava, että tämä kerääminen tapahtuu laillisesti ja läpinäkyvästi. Käyttäjille on tarjottava selkeä tieto siitä, miten heidän tietonsa kerätään ja käytetään sekä heidän on annettava suostumuksensa tähän prosessiin. Tietojen on myös oltava tarkkoja ja ajantasaisia, ja niitä on päivitettävä tarvittaessa. Lisäksi henkilötietoja ei saa säilyttää pidempään kuin on tarpeen, ja niiden on oltava

suojattuja eheän ja luottamuksellisen käsittelyn takaamiseksi. Lopuksi yrityksen on oltava vastuussa henkilötietojen asianmukaisesta käsittelystä ja dokumentoitava tämän noudattaminen. Ymmärtämällä ja noudattamalla näitä GDPR:n periaatteita yritykset voivat varmistaa, että niiden toiminta on lainmukaista ja että ne suojaavat asiakkaidensa tietoja asianmukaisesti. (Özkan, 2023)

2.6 Harhaanjohtava visualisointi

Datan visualisoinnista on paljon hyötyä, kuten aiemmin on jo todettu, mutta siitä löytyy myös haittapuolia. Yksi näistä haittapuolista on harhaanjohtava visualisointi. Harhaanjohtavassa visualisoinnissa ei välttämättä suoraan valehdella lukijalle, vaan kaavio tai muu vastaava on tehty ja asetettu niin, että nopealla vilkaisulla lukija ymmärtää kaavion väärin, kuten esimerkiksi kuvassa 4, jossa ero vaikuttaa visuaalisesti paljon suuremmalta, kuin se oikeasti on. Kuvassa kerrotaan The Times-sanomalehden voittavan Daily Telegraphin myynnissä, vaikkakin se pitää paikkansa, oikealla olevasta pylväskaaviosta saa käsityksen, että The Timesin myynti on melkein kaksinkertaista Daily Telegraphiin verrattuna (kuva 6). Oikeasti ero ei ole kuin 10 %. Harhaanjohtavaa tilastojen käyttöä nähdään usein mainonnassa, politiikassa, mediassa ja muilla yhteiskunnan alueilla (Calzon, 2023).

Kuva 4. Harhaanjohtava visualisointi.



3 Python visualisointityökaluna

Python on ohjelmointikielenä hyvin suosittu ja yksinkertainen, joten se on hyvä valinta aloittelijalle. Pythonia käytetään kaikenlaiseen ohjelmointiin. Esimerkiksi data-analytiikkaan ja koneoppimiseen, web-kehitykseen, automatisointiin ja skriptaukseen, ohjelmistotestaukseen ja prototyyppeihin sekä jonkun verran työpöytäsovellusten kehittämiseen. (Dillemuth, 2022)

Python-ohjelmointikieli on suosittu monien ihmisten keskuudessa. Se ilmestyi ensimmäisen kerran vuonna 1991 ja siitä lähtien se on kasvanut yhdeksi suosituimmista ohjelmointikielistä. Erityisesti viimeisen parinkymmenen vuoden aikana Python on tullut erittäin suosituksi erilaisten verkkosivustojen rakentamisessa käytettävien kehysten, kuten Railsin (Ruby) ja Djangon (Python) avulla. Pythonia ja muita vastaavia kieliä kutsutaan usein skriptikieliksi, sillä niitä voidaan käyttää nopeasti pienten ohjelmien tai skriptien kirjoittamiseen automatisoimaan erilaisia tehtäviä.

Termi "skriptikieli" ei kuitenkaan aina anna oikeaa kuvaa näiden kielten monipuolisista käyttömahdollisuuksista. Pythonilla on vahva tieteellisen laskennan ja data-analyysiyhteisö, ja se on noussut tärkeäksi kieleksi datatieteessä, koneoppimisessa ja yleisessä ohjelmistokehityksessä.

Pythonia verrataan usein muihin ohjelmointikieliin, kuten R:ään, MATLABiin ja muihin, kun puhutaan data-analyysistä ja vuorovaikutteisesta laskennasta. Viime vuosina Pythonin avoimen lähdekoodin kirjastot, kuten pandas ja scikit-learn, ovat tehneet siitä suositun valinnan data-analyysitehtäviin. Yhdistettynä Pythonin yleiseen vahvuuteen yleiskäyttöisen ohjelmistotekniikan alalla, se on erinomainen valinta datan sovellusten rakentamiseen. (McKinney, 2017)

3.1 Matplotlib ja seaborn

Matplotlib on yleisesti käytetty Python-kirjasto, joka on suunniteltu kaavioiden ja muiden kaksiulotteisten datavisualisointien luomiseen. Sen on luonut alun perin neurobiologi John D.Hunter (1968-2012) jatkokoulutuksenaan neurobiologiassa visualisoidakseen epilepsiapotilaiden elektrokortikografian dataa. Nykyään se on laajan kehittäjätiimin ylläpitämä. Vaikka Python-ohjelmoijilla on nykyään useita visualisointikirjastoja käytössään, silti matplotlib on erittäin suosittu ja integroituu loistavasti muihin Pythonin työkaluihin.

Matplotlib alkoi Hunterin aloittamana projektina vuonna 2002. Tarkoituksena oli, että Pythonissa voitaisiin tehdä kaavioita samalla tavalla kuin MATLABissa. MATLAB eli Matrix

Laboratory on yksi suosituimmista ja laajalti käytetyistä teknisistä laskentaohjelmista. Se tarjoaa monipuolisia työkaluja ja toimintoja matriisien käsittelyyn, numeeriseen laskentaa, signaalikäsittelyyn, kuvankäsittelyyn, simulointiin, optimointiin ja se myös sisältää graafisia työkaluja visualisointien luomiseen ja interaktiiviseen ohjelmointiin (Warren, 2015). Matplotlib- ja IPython-yhteisöt ovat tehneet yhteistyötä, jotta visualisointi olisi helpompaa IPython-ympäristössä ja nykyään myös Jupyter Notebookissa. Matplotlib toimii eri käyttöjärjestelmien käyttöliittymien kanssa ja pystyy muodostamaan kaavioita eri tiedostomuotoihin, kuten PDF, SVG, JPG, PNG, BMP ja GIF (McKinney, 2017).

Vaikka Matplotlib on todistetusti kätevä ja suosittu työkalu visualisointia varten, se jättää kuitenkin joskus hyvin paljon toivomisen varaan monien käyttäjien mukaan. Matplotlibin ohjelmointirajapinta eli API (Application Programming Interface) on melko matalalla tasolla. Matplotlib vaatii usein paljon koodin runkoa tehdäkseen monimutkaisia tilastollisia visualisointeja. Matplotlib on kehitetty ennen Pandasia eikä sitä ole sen takia suunniteltu käytettäväksi sen kanssa. Pandas on Pythonin avoimen lähdekoodin kirjasto, joka tarjoaa tehokkaan ja joustavan tavan työstää rakenteellista dataa. Se sisältää DataFrame tietorakenteen, joka on tehokas työkalu data-analyysiin ja manipulointiin. Seaborn kehitettiin näitä ongelmia varten. Se tarjoaa API.n Matplotlibin päälle, joka antaa järkeviä valintoja kaavioiden tyyliksi ja väreiksi, määrittelee yksinkertaisia korkean tason toimintoja yleisille tilastollisille visualisoinneille ja integroituun Pandas Dataframeen. (McKinney, 2017)

3.2 Plotly

Plotly on Montrealilainen tekoälyyn ja analytiikkaan erikoistunut yritys. Heidän pääpainonsa on analytiikkatyökalujen kehittämisessä, mutta he ovat myös julkaisseet ilmaisen ja avoimen lähdekoodin kirjaston nimeltään Plotly Pythonille, R:lle, Matlabille sekä Julialle (VanderPlas, 2017)

Plotlyn yksi merkittävimmistä ominaisuuksista on sen kyky tuottaa kaavioita sekä staattisina että interaktiivisina versioina. Plotly tarjoaa myös laajan valikoiman kaaviointivaihtoehtoja, kuten perus- ja tilastolliset kaaviot, kartat, 3D-kaaviot, alakaaviot ja niin edelleen. Plotlyn interaktiivisiin ominaisuuksiin kuuluu esimerkiksi zoomaus, vetäminen ja ponnahdusikkunat. Plotlyn kaavioita voidaan helposti upottaa esimerkiksi nettisivuille. Plotly on yhteensopiva monien muiden Python kirjastojen kanssa. Niistä suurimpina esimerkkeinä Pandas ja NumPy. NumPy on lyhyesti yksi Pythonin keskeisimmistä kirjastoista numeerisen laskennan toteuttamiseen. Se on luotu toteuttamaan monipuolisia matemaattisia ongelmia, kuten esimerkiksi matriisien käsittelyä, lineaarista algebraa, satunnaislukujen generointia ja niin edelleen (Majumder, 2021).

Vuonna 2017 Plotly julkaisi beetaversion Pythonilla kirjoitettavasta web-kehyksestään Dashista ja vuonna 2019 sen koko versio julkaistiin virallisesti. Vuonna 2019 luotiin Dash for R, ja vuonna 2020 Dash for Julia, mikä johti yleisimpien datatiede-ohjelmakielten pohjalta luotuun kieliriippumattomaan kehykseen. Dash on kirjoitettu Flaskin, Plotly.js:n ja React.js:n päälle, joka mahdollistaa interaktiivisten kaavioiden luomisen tavallisella Pythonilla ilman taakkaa taustakoodista, reiteistä ja pyynnöistä. Dash renderöidään selaimessa, mikä tekee siitä monialustaisen ratkaisun. Markkinoilta löytyy 2 eri versioita Dash-kehyksestä. ”Dash Open Source” ja ”Dash enterprise”. Ensin mainittu on nimensä mukaisesti avoimen lähdekoodin ja ilmaiseksi saatavilla oleva kehys ja jälkimmäinen on kaupallinen versio, joka tarjoaa lisäominaisuuksia ja työkaluja ja on sijoiteltu suurempien yritysten ja organisaatioiden tarpeisiin. (Kilcommins, 2021).

3.3 Dashin Callback-funktio

Plotlyn Dash-kehys käyttää callback-funktioita päivittääkseen sovelluksen komponentteja käyttäjä vuorovaikutuksen perusteella. Callback-funktio on Python-funktio, joka suoritetaan automaattisesti, kun käyttäjä esimerkiksi klikkaa painiketta tai muuttaa valikon arvoa. Tämä funktio ottaa syötteenä yhden tai useamman komponentin tilan, kuten painikkeen klikkausten määrä esimerkiksi, ja palauttaa uuden arvon, joka päivittää komponentin tilan.

Callback-funktioiden avulla voidaan luoda dynaamisia ja interaktiivisia sovelluksia, joissa komponentit reagoivat reaaliaikaisesti käyttäjän toimintaan. Tämä tekee Dash-sovelluksista erittäin joustavia ja käyttäjäystävällisiä. (Plotly, 2024e)

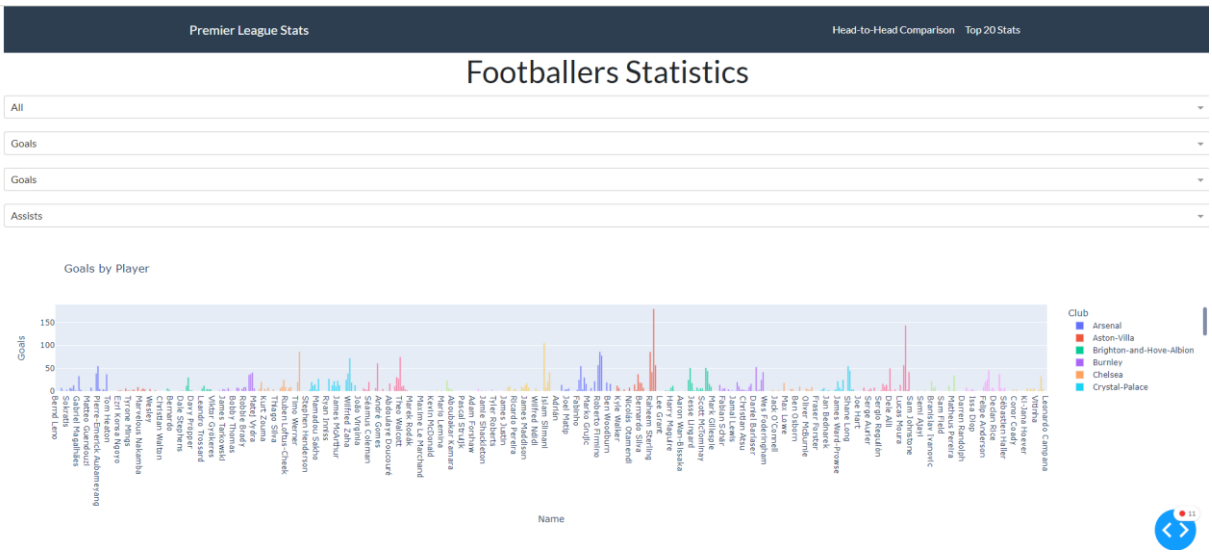
4 Sovellus

Opinnäytetyön käytännön osuudessa on tarkoitus luoda sovellus, joka visualisoi Englannin Valioliigan joukkueiden ja pelaajien tilastoja. Data on haettu Kagglesta, ja aineistona on käytetty English Premier League -nimistä tietoaineistoa. Tietoaineisto on CSV-muodossa, mikä tekee sen helposti ladattavaksi ja käytettäväksi sovellusta varten.

Ensimmäiseksi täytyy ladata tarvittavat kirjastot. Näihin kirjastoihin kuuluu tärkeimpänä sovelluksen rakentamista varten Dash, jonka jälkeen tulevat sen komponentit: Dash Bootstrap Components, Dash Core Components ja Dash HTML Components. Dash Bootstrap Components -kirjastoa käytetään Bootstrap-teeman tuomiseen Dash-sovellukseen, jotta sovelluksesta saadaan visuaalisesti miellyttävämpi (Dash Bootstrap Components, 2024). Dash Core Components ja Dash HTML Components -kirjastoja käytetään sovelluksen komponenttien, kuten graafien, pudotusvalikoiden ja muiden HTML-elementtien luomiseen (Plotly, 2024). Sovellukseen ladataan myös Pandas-kirjasto, jota käytetään tietojen käsittelyyn, kuten lukemiseen, suodattamiseen ja analysointiin (Pandas, 2024). Viimeiseksi ladataan Plotly Express, jota käytetään graafien luomiseen (Plotly, 2024a).

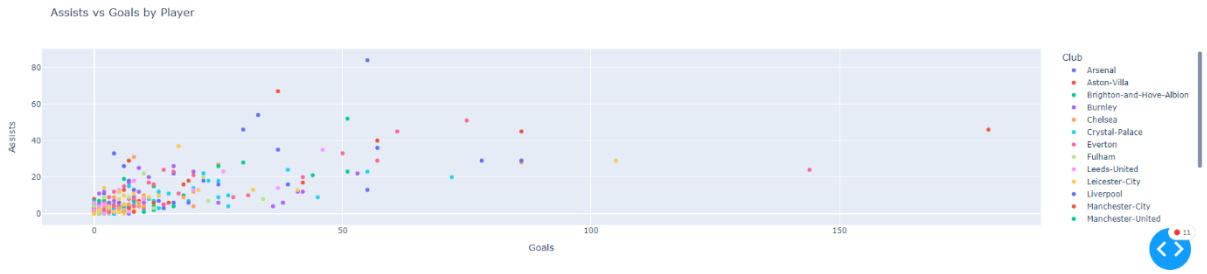
Sovelluksessa on kolme eri sivua. Ensimmäinen on kuvassa 5 esiintyvä pääsivu "Premier League Stats", joka sisältää pylväskaavion ja pistekaavion. Pylväskaaviota voidaan suodattaa valitsemalla tilasto, jota halutaan tarkastella sekä vain tietyn joukkueen pelaajat.

Kuva 5. Pääsivu ja pylväskaavio



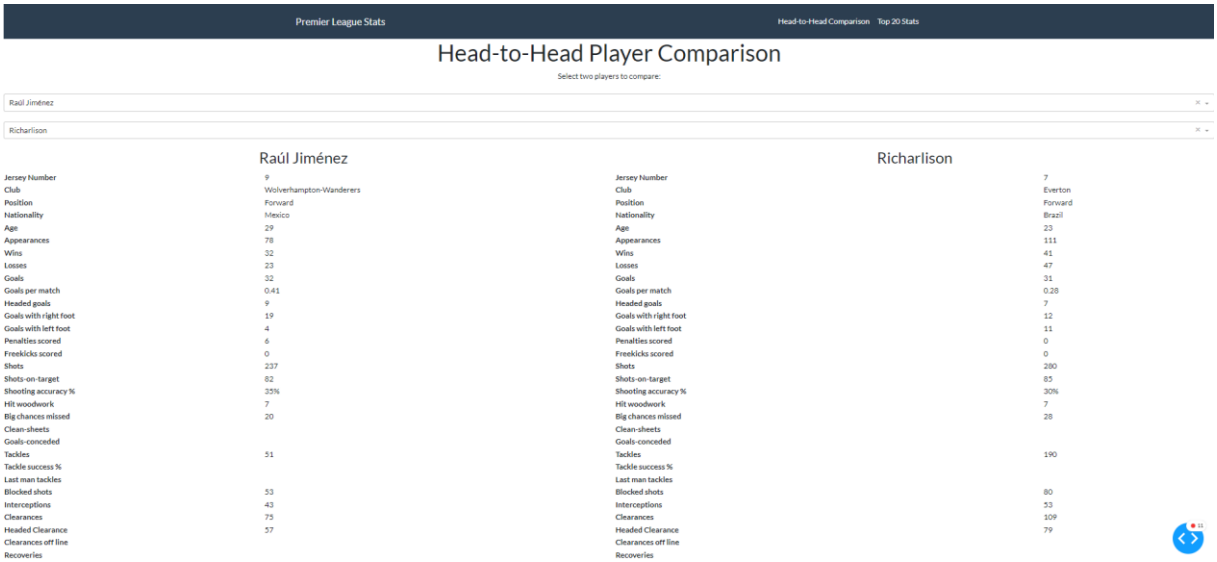
Kuvassa 6 olevan pistekaavion avulla voidaan tarkastella kahden eri tilaston suhdetta keskenään. Valitut tilastot voi olla esimerkiksi maalit ja syötöt ja pistekaaviota suodattuu pylväskaavion tapaan myös joukkueen mukaan.

Kuva 6. Pääsivun pistekaavio.



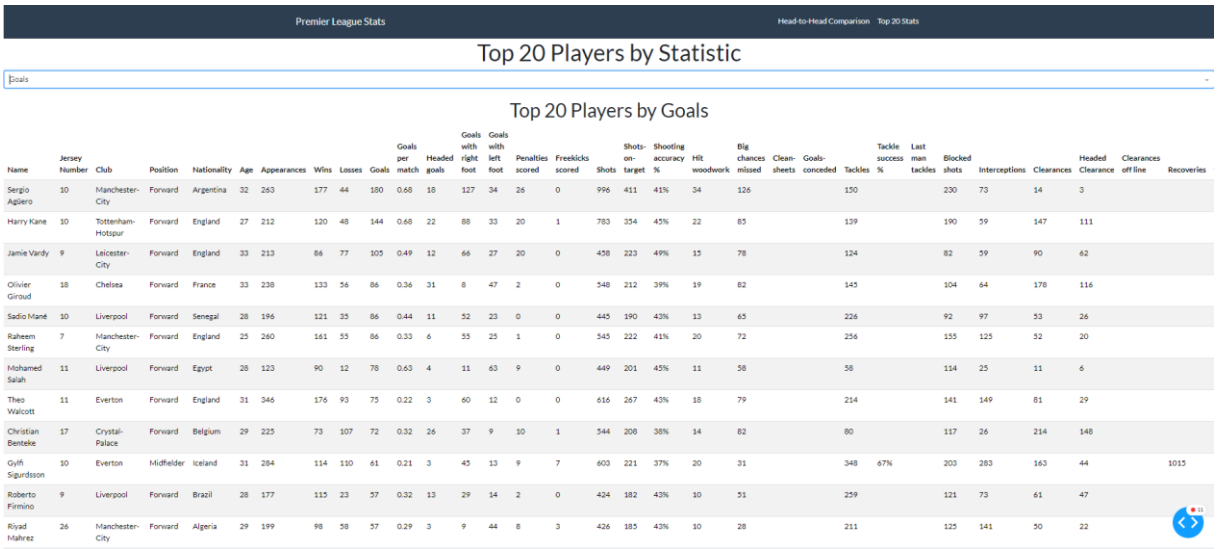
Toinen sivu on kuvan 7 "Head-to-Head comparison", joka on tarkoitettu kahden pelaajan keskinäiseen vertailuun. Sivulla voidaan valita pudotusvalikoiden avulla kaksi eri pelaajaa. Kun pelaajat on valittu, sivulle ilmestyy heidän tietonsa ja tilastonsa vierekkäin, jotta niiden vertailu olisi yksinkertaista ja vaivatonta.

Kuva 7. Vertailusivu



Kolmas sivu on kuvan 8 "Top 20 stats", joka näyttää 20 parasta pelaajaa käyttäjän valitseman tilaston perusteella. Esimerkiksi, jos valittu tilasto on maalit näkyy 20 parasta maalintekijää tai torjunnat niin ilmestyy 20 eniten torjuntoja tehnyttä ja niin edelleen.

Kuva 8. Top-20 sivu.



4.1 Tilastojen lataaminen ja sovelluksen asetusten määrittäminen

Seuraavaksi hankitaan CSV-tiedosto, joka sisältää Englannin Valioliigan kauden 2020-21 pelaajien tilastot. Kuvassa 9 tiedosto tallennetaan "data"-nimiseen DataFrame-taulukkoon, jotta sitä voidaan käyttää tulevissa osiossa helposti.

Itse sovellus luodaan käyttämällä "app"-muuttujaa, johon kutsutaan Dash-luokan konstruktori "dash.Dash()". Kun konstruktori kutsutaan, se alustaa uuden sovelluksen, johon voidaan välittää erilaisia parametreja, jotka määrittävät sovelluksen käyttäytymistä ja ulkoasua. Tässä sovelluksessa käytetään "external_stylesheet"-parametria, jolla sovellukseen tuodaan Bootstrap-teema "FLATLY", jonka Dash lataa ja liittää osaksi sovelluksen HTML-koodia.

Kuva 9. Datasetin ja Bootstrap-teeman tuonti.

```
# Load the dataset
data = pd.read_csv('premierleague-2020-09-24.csv')

# Initialize the Dash app with Bootstrap
app = dash.Dash(__name__, external_stylesheets=[dbc.themes.FLATLY])
app.title = "Premier League Stats"
```

4.2 Navigaatiopalkin määrittäminen

Sovelluksen sivujen väliseen navigointiin on luotu yksinkertainen navigointipalkki käyttäen "dbc.NavbarSimple"-komponenttia, joka on Dash Bootstrap Components -kirjaston yksinkertaistettu navigaatiopalkkikomponentti kuvassa 10. "Brand"-parametri määrittää navigointipalkin vasempaan reunaan sijoitettavan brändin tai otsikon tekstin, joka on yleensä sovelluksen tai verkkosivun nimi. "Brand_href"-parametri määrittää linkin URL-osoitteen, johon otsikkoteksti vie käyttäjän – tässä tapauksessa etusivulle. "Children"-parametri määrittää navigointipalkin lapsielementit, jotka vievät sovelluksen kahdelle muulle sivulle (Dash Bootstrap Components, 2024a).

Kuva 10. Navigaatiopalkin koodi.

```
# Navigation bar
navbar = dbc.NavbarSimple(
    brand="Premier League Stats",
    brand_href="/",
    color="primary",
    dark=True,
    children=[
        dbc.NavItem(dbc.NavLink("Head-to-Head Comparison", href="/comparison")),
        dbc.NavItem(dbc.NavLink("Top 20 Stats", href="/top-20")),
    ]
```

4.3 Tilastovaihtoehdot pudotusvalikkoa varten

Tilastojen tarkasteluun tarvittavat vaihtoehdot on tallennettu kuvan 11 "stats_options"-listaan, joka sisältää sanakirjoja. Yksi sanakirja edustaa yhtä tilastoa ja määrittelee "label" ja "value" -parin.

"Label" määrittää tekstin, jonka käyttäjä näkee pudotusvalikossa. Jos label on esimerkiksi "goals", käyttäjä näkee valikossa "goals"-vaihtoehdon ja voi sen valitessaan tarkastella maalimääriä. "Value" puolestaan määrittää sen arvon, jonka Dash etsii CSV-tiedostosta ja käyttää viitatakseen tiettyyn tilastoon. Toisin sanoen value vastaa CSV-tiedoston sarakkeen nimiä. Myöhemmin sovelluksen eri sivuilla tätä "stats_options"-listaa käytetään välittämällä se Dash-komponenteille, kuten esimerkiksi pudotusvalikossa.

Kuva 11. Tilastovaihtoehdot

```
stats_options = [
    {'label': 'Appearances', 'value': 'Appearances'},
    {'label': 'Goals', 'value': 'Goals'},
    {'label': 'Assists', 'value': 'Assists'},
    {'label': 'Shots', 'value': 'Shots'},
    {'label': 'Shots-on-target', 'value': 'Shots-on-target'},
    {'label': 'Clean-sheets', 'value': 'Clean-sheets'},
    {'label': 'Goals-conceded', 'value': 'Goals-conceded'},
    {'label': 'Interceptions', 'value': 'Interceptions'},
    {'label': 'Tackles', 'value': 'Tackles'},
    {'label': 'Clearances', 'value': 'Clearances'},
    {'label': 'Own-goals', 'value': 'Own-goals'},
    {'label': 'Saves', 'value': 'Saves'},
    {'label': 'Yellow-cards', 'value': 'Yellow-cards'},
    {'label': 'Red-cards', 'value': 'Red-cards'},
```

4.4 Pääsivun asettelu ja kaaviot

Dash-sovelluksen asettelu määritetään "layout"-attribuutilla, joka sisältää komponentit, jotka muodostavat sovelluksen käyttöliittymän. Pääsivun asettelu on sisällytetty "html.Div"-komponenttiin kuvassa 12. Tämä yleinen komponentti toimii säilönä muille komponenteille ja mahdollistaa sovelluksen elementtien selkeän ryhmittelyn (Plotly, 2024b). Aikaisemmin määriteltyä navigointipalkkia hyödynnetään nyt "navbar"-elementtinä, joka on lisätty osaksi pääsivun asettelua. Sen avulla käyttäjä voi vaihdella eri sivujen välillä.

Kuva 12. Pääsivun asettelu.

```
# Layout for the main statistics page
main_layout = dbc.Container([
    navbar,
    dbc.Row([
        dbc.Col([
            html.H1("Footballers Statistics", className='text-center'),
```

Sivulle on tehty neljä eri pudotusvalikkoa kuvassa 13. Ensimmäinen on joukkuevalikko, josta käyttäjä voi valita tarkastellaanko koko sarjan vai yksittäisen joukkueen tilastoja. Toinen pudotusvalikko on tarkoitettu pylväskaaviolle. Kaksi muuta pudotusvalikkoa on tehty pistekaavion akseleita varten, toinen y-akselia ja toinen x-akselia varten.

Kuva 13. Pudotusvalikot.

```
dcc.Dropdown(
    id='club-dropdown',
    options=[{'label': 'All', 'value': 'All'}] + [{'label': club, 'value': club} for club in data['Club'].unique()],
    value='All',
    clearable=False,
    style={'margin-bottom': '20px'}
),
dcc.Dropdown(
    id='statistic-dropdown',
    options=stats_options,
    value='Goals',
    clearable=False,
    style={'margin-bottom': '20px'}
),
dcc.Dropdown(
    id='x-axis-dropdown',
    options=stats_options,
    value='Goals',
    clearable=False,
    style={'margin-bottom': '20px'}
),
dcc.Dropdown(
    id='y-axis-dropdown',
    options=stats_options,
    value='Assists',
    clearable=False,
    style={'margin-bottom': '20px'}
```

Pudotusvalikot sisältävät useita komponentteja. Kuva 14 esiintyvä "id" on uniikki tunniste, jonka avulla Dash voi viitata komponenttiin callback-funktiossa. "Options" käyttää aikaisemmin määriteltyä "stats_options"-listaa, joka luo pudotusvalikkoon tilastovaihtoehdot. "Value" on oletusarvoksi valittu tilasto, ja "clearable=False" estää käyttäjää tyhjentämästä valintaa, jolloin yksi tilasto on aina valittuna. "dcc.Graph" on Dash-komponentti kaavioiden ja graafien esittämistä varten. Sovelluksen kaaviot sisältävät kukin uniikin "id"-tunnisteen callback-funktioita varten (Plotly, 2024c).

Kuva 14. Kaavioiden id-tunnisteet.

```
dbc.Row([
    dbc.Col(dcc.Graph(id='stats-bar-chart'), width=12),
]),
dbc.Row([
    dbc.Col(dcc.Graph(id='stats-scatter-chart'), width=12),
]),
```

4.5 Vertailusivun asettelu

Kuvassa 15 on sovelluksen toinen sivu eli vertailusivu, joka on suunniteltu kahden pelaajan tilastojen keskinäiseen vertailuun. Sivun tarkoituksena on tarjota käyttäjälle mahdollisuus tarkastella valitsemiensa pelaajien suorituskykyä rinnakkain.

Sivun rakenteessa on ensin otsikko, joka informoi käyttäjää sivun tarkoituksesta ja toiminnallisuudesta. Tämän jälkeen sivulla on kaksi pudotusvalikkoa, joiden avulla käyttäjä voi valita vertailtavat pelaajat koko Valioliigan pelaajien joukosta.

Kun käyttäjä on valinnut molemmat pelaajat, sivulle ilmestyy heidän tilastonsa rinnakkain. Tämä mahdollistaa helpon ja nopean vertailun pelaajien suorituskyvystä eri tilastojen osalta. Vertailu voi sisältää esimerkiksi maalimäärät, syötöt, peliminuutit tai muut oleelliset pelaajakohtaiset tilastot.

Sivun suunnittelussa on kiinnitetty huomiota käyttäjäystävällisyyteen ja selkeään informaation esittämiseen, jotta pelaajien välinen vertailu olisi mahdollisimman vaivatonta ja informatiivista.

Kuva 15. Vertailusivun asettelun koodi.

```
# Layout for the head-to-head comparison page
comparison_layout = dbc.Container([
    navbar,
    dbc.Row([
        dbc.Col([
            html.H1("Head-to-Head Player Comparison", className='text-center'),
            html.Div("Select two players to compare:", className='text-center mb-4'),
            dcc.Dropdown(
                id='player1-dropdown',
                options=[{'label': player, 'value': player} for player in data['Name'].unique()],
                placeholder='Select Player 1',
                style={'margin-bottom': '20px'}
            ),
            dcc.Dropdown(
                id='player2-dropdown',
                options=[{'label': player, 'value': player} for player in data['Name'].unique()],
                placeholder='Select Player 2',
                style={'margin-bottom': '20px'}
            ),
            html.Div(id='comparison-content')
        ], width=12)
    ])
], fluid=True)
```

Kummallekin pudotusvalikolle on annettu oma ”id” myöhemmin käsiteltäviä vastakutsu-funktioita varten. Pudotusvalikko on hyvin yksinkertainen ja siitä pystyy ainoastaan valitsemaan aineistosta löytyviä pelaajia ja paikkamerkissä pyydetään käyttäjää valitsemaan pelaajat vertailua varten.

4.6 Top-20 sivun asettelu

Kuvassa 16 esiintyvä sovelluksen kolmas sivu on suunniteltu näyttämään 20 parasta pelaajaa käyttäjän valitsemassa tilastossa. Sivun tarkoituksena on tarjota nopeasti ja selkeästi tietoa Valioliigan kärkipelaajista eri osa-alueilla.

Käyttäjä voi pudotusvalikosta valita tarkasteltavan tilaston, kuten maalit, syötöt, keltaiset kortit tai torjunnat. Kun tilasto on valittu, sivulle ilmestyy lista 20 pelaajasta, joilla on korkein arvo kyseisessä tilastossa. Tämä mahdollistaa nopean vertailun ja yleiskuvan saamisen Valioliigan huippupelaajista.

Sivun asettelu on tarkoituksella pidetty yksinkertaisena ja selkeänä. Pudotusvalikko noudattaa samaa rakennetta kuin sovelluksen muissa osioissa käytetyt valikot, mikä takaa yhtenäisen käyttökokemuksen koko sovelluksessa. Käyttäjä voi helposti vaihtaa tarkasteltavaa tilastoa ja nähdä välittömästi päivitetyn listan parhaista pelaajista.

Kuva 16. Top-20 sivun asettelun koodi

```
# Layout for the top 20 statistics page
top_20_layout = dbc.Container([
    navbar,
    dbc.Row([
        dbc.Col([
            html.H1("Top 20 Players by Statistic", className='text-center'),
            dcc.Dropdown(
                id='top-stat-dropdown',
                options=stats_options,
                value='Goals',
                clearable=False,
                style={'margin-bottom': '20px'}
            ),
            html.Div(id='top-20-content')
        ], width=12)
    ])
], fluid=True)
```

5 Reititys ja sivunvaihto

Sovelluksen reititys perustuu kahteen pääkomponenttiin: "dcc.Location" ja "display_page"-funktioon, jotka esiintyvät kuvassa 17. "Dcc.Location"-komponentti on Dashin sisäinen työkalu, joka seuraa selaimen URL-osoitetta. Se sisältää "id=url" -tunnisteen, jota käytetään sovelluksessa URL-tietojen hakemiseen. Komponenttiin on lisätty "refresh=false", mikä estää sivun uudelleenlataamisen selaimessa ja mahdollistaa sivun sisällön dynaamisen päivittämisen ilman koko sivun uudelleenlataamista.

"Display_page"-funktio vastaa oikean sivun näyttämisestä URL-osoitteen perusteella. Se saa argumenttina "pathname"-polun ja palauttaa sen perusteella oikean sivun asettelun. "Output('page-content', 'children')" määrittää, että funktion palauttama arvo asetetaan elementtiin, jonka "ID" on "page-content". Tämän elementin sisältö muuttuu aina URL-polun vaihtuessa.

"Input('url', 'pathname')" kertoo, että funktio aktivoituu aina, kun "dcc.Location"-komponentti havaitsee muutoksen URL-osoitteessa. URL-polku (esimerkiksi "/compare" tai "/top-20") välitetään "display_page"-funktiolle argumenttina (Plotly, 2024d).

Kuva 17. Sovelluksen reititys.

```
app.layout = html.Div([
    dcc.Location(id='url', refresh=False),
    html.Div(id='page-content')
])

@app.callback(
    Output('page-content', 'children'),
    [Input('url', 'pathname')]
)
def display_page(pathname):
    if pathname == '/comparison':
        return comparison_layout
    elif pathname == '/top-20':
        return top_20_layout
    else:
        return main_layout
```

5.1 Kaavioiden päivitys pääsivulla

Dash-sovelluksissa callback-funktiot mahdollistavat elementtien päivittämisen käyttäjän toimien perusteella ilman sivun uudelleenlataamista. Tässä sovelluksessa kaavioiden päivitys perustuu käyttäjän valintoihin jalkapalloseuran ja tilastojen osalta.

Kuvan 18 "@app.callback" osoittaa, että kyseessä on callback-funktio. Output-funktiota käytetään määrittämään päivitettävät elementit, joita ovat tässä tapauksessa pylväsdiagrammi ("bar-chart") ja pistekaavio ("scatter-plot"). Input-parametrit määrittävät callback-funktion aktivoitumisen käyttäjän valitessa seuran "-club-dropdown"-, "x-axis-dropdown"- tai "y-axis-dropdown"-pudotusvalikosta. Kun käyttäjä valitsee uuden seuran tai muuttaa tilastoja x- tai y-akselilla, kaaviot päivittyvät automaattisesti (Plotly, 2024d).

Datan suodattamiseen käytetään "filtered_data"-funktiota, joka määrittää, onko kaikki vai joku tietty seura valittuna kaavioita varten. Funktiot "px.bar" ja "px.scatter" hyödyntävät Plot Expressiä (px) kaavioiden tyypin määrittämiseen. Näiden funktioiden sisällä määritetään kaavioiden ominaisuudet, kuten otsikot ja akseleille valitut arvot. Lopuksi "return"-funktiolla callback-funktio palauttaa pylväsdiagrammin ja pistekaavion.

Kuva 18. Kaavioiden päivitys pääsivulle.

```
@app.callback(
    [Output('stats-bar-chart', 'figure'),
     Output('stats-scatter-chart', 'figure')],
    [Input('club-dropdown', 'value'),
     Input('statistic-dropdown', 'value'),
     Input('x-axis-dropdown', 'value'),
     Input('y-axis-dropdown', 'value')]
)
def update_main_page(selected_club, selected_stat, x_stat, y_stat):
    filtered_data = data if selected_club == 'All' else data[data['Club'] == selected_club]

    bar_fig = px.bar(filtered_data, x='Name', y=selected_stat, color='Club',
                     title=f'{selected_stat} by Player{" for " + selected_club if selected_club != "All" else ""}')

    scatter_fig = px.scatter(filtered_data, x=x_stat, y=y_stat, color='Club',
                             hover_data=['Name'],
                             title=f'{y_stat} vs {x_stat} by Player')

    return bar_fig, scatter_fig
```

5.2 Vertailusivun päivittäminen

Kuvassa 19 vertailusivun päivittämiseen käytetään samaa Dashin callback-funktiota kuin pääsivunkin päivittämiseen. Tällä kertaa output-funktioon on valittu päivitettäväksi elementti, jonka ID on "comparison-content". Elementin "children" määrittää HTML-sisällön kyseiselle elementille. Input-funktioihin on valittu molempia pelaajia varten olevat pudotusvalikoiden elementit, ja funktio aktivoituu aina, kun käyttäjä valitsee pelaajan ensimmäisestä tai toisesta valikosta. Näiden avulla sovellus saa tietää, keitä kahta pelaajaa käyttäjä haluaa vertailla.

Funktion sisälle on lisätty if-sääntö, joka tarkistaa, että molemmista valikoista on valittuna jokin pelaaja. Mikäli näin ei ole, sovellus palauttaa tekstin "Please select two players to compare". Kun käyttäjä on valinnut kaksi pelaajaa, sovellus hakee heidän tietonsa datasta. Koska jokaisella pelaajalla on vain yksi rivi tilastoja, otetaan ensimmäinen rivi käyttämällä ".iloc[0]"-indeksiä.

Kun molemmat tilastot on haettu, "player_stats"-funktio hoitaa taulukon luomisen yksittäiselle pelaajalle. "Html.Table" luo HTML-taulukon, joka koostuu riveistä "html.Tr" ja soluista, jotka sisältävät tilaston otsikon "html.Th" ja sen arvon "html.Td". "For col in data.columns if col != 'Name'"-funktio käy läpi kaikki sarakkeet DataFramessa (esimerkiksi maalit, syötöt, taklaukset) ja luo jokaisesta sarakkeesta uuden taulukkorivin, paitsi "Name"-sarakkeesta, joka on jo pelaajan nimi.

Kun molempien pelaajien tilastot on haettu ja taulukot luotu, palautetaan nämä tiedot vierekkäin "dbc.Row"-jaon avulla. "Dbc.Row" ja "Dbc.Col" ovat Dash Bootstrap -komponentteja, jotka mahdollistavat sisällön jakamisen riveihin ja sarakkeisiin. Tässä tapauksessa molemmat pelaajat esitetään omassa sarakkeessaan (leveys = 6, joka tarkoittaa 50 % tilasta) (Dash Bootstrap Components, 2024).

Kuva 19. Vertailusivun päivitys.

```python
@app.callback(
    Output('comparison-content', 'children'),
    [Input('player1-dropdown', 'value'),
     Input('player2-dropdown', 'value')]
)
def update_comparison(player1, player2):
    if not player1 or not player2:
        return html.Div(children='Please select two players to compare.', className='text-center text-danger')

    player1_data = data[data['Name'] == player1].iloc[0]
    player2_data = data[data['Name'] == player2].iloc[0]

    def player_stats(player_data):
        return html.Table([
            html.Tr([html.Th(col), html.Td(player_data[col])]) for col in data.columns if col != 'Name'
        ], className='table table-striped')

    return dbc.Row([
        dbc.Col([
            html.H3(player1, className='text-center'),
            player_stats(player1_data)
        ], width=6),
        dbc.Col([
            html.H3(player2, className='text-center'),
            player_stats(player2_data)
        ], width=6),
    ])
```

5.3 Top 20-sivun päivittäminen

Kuvan 20 callback-funktiossa päivitetään sivun sisältö näyttämään 20 parasta pelaajaa valitun tilaston perusteella. Kun käyttäjä valitsee tilaston pudotusvalikosta, sovellus hyödyntää pandas-kirjaston "nlargest"-metodia hakeakseen 20 parasta pelaajaa kyseisessä tilastossa (Pandas, 2024a).

Funktio muodostaa taulukon, jossa esitetään pelaajien tiedot ja valitun tilaston arvot. Taulukon otsikkorivillä näkyvät sarakkeiden nimet, ja itse data esitetään riveittäin. Bootstrapin tyylit tuovat taulukkoon raidallisen visuaalisen ilmeen, mikä helpottaa datan hahmottamista. Tämä mahdollistaa käyttäjälle eri tilastojen helpon vertailun ja 20 parhaan pelaajan tarkastelun kustakin tilastosta.

Kuva 20. Top 20-sivun päivitys.

```python
@app.callback(
    Output('top-20-content', 'children'),
    [Input('top-stat-dropdown', 'value')]
)
def display_top_20(selected_stat):
    top_20_data = data.nlargest(20, selected_stat)

    return html.Div([
        html.H2(f"Top 20 Players by {selected_stat}", className='text-center'),
        html.Table([
            html.Thead(html.Tr([html.Th(col) for col in top_20_data.columns])),
            html.Tbody([
                html.Tr([html.Td(top_20_data.iloc[i][col]) for col in top_20_data.columns]) for i in range(len(top_20_data))
            ])
        ], className='table table-striped')
    ])
```

6 Tulokset

Opinnäytetyön teoriaosassa keskityttiin datan visualisoinnin perusteisiin ja eri visualisointityökalujen vertailuun. Tutkimuksessa havaittiin, että datanvisualisointi on olennainen osa tiedon ja analysointia ja viestintää. Visualisoinnin avulla monimutkainen tietomäärä voidaan esittää selkeästi ja ymmärrettävästi graafisessa muodossa, mikä tehostaa käyttäjien kykyä hahmottaa ja tulkita tietoa. Visualisointityökaluja tutkimalla huomattiin, että Pythonin kirjastot Matplotlib ja Seaborn ovat hyviä työkaluja staattisten visualisointien luomiseen, kun taas työssä käytetty Plotly erottuu erityisesti interaktiivisilla ja dynaamisilla kaavioillaan. Plotlyn interaktiiviset ominaisuudet, kuten esimerkiksi zoomaus, vetäminen ja ponnahdusikkunat tekevät datan tarkastelusta käyttäjäystävällistä ja joustavaa.

Käytännönosan tuloksena syntyi hyvin selkeä ja käyttäjäystävällinen Dash-sovellus, jonka avulla käyttäjä voi vertailla ja tarkkailla Englannin Valioliigan pelaajien tilastoja. Plotlyn Dash-kehys osoittautui erinomaiseksi työkaluksi interaktiivisten ja dynaamisten sovellusten tekoon.

7 Johtopäätökset ja pohdinta

Datan visualisointi on nykyaikaisessa tietoanalytiikassa keskeinen työkalu, joka muuntaa monimutkaiset tietomassat helposti ymmärrettäviksi kokonaisuuksiksi. Opinnäytetyöprosessin aikana kävi selväksi, että oikein toteutettuna visualisointi voi merkittävästi parantaa tiedon tulkintaa ja päätöksentekoa.

Python-kirjastojen vertailussa Plotly erottui edukseen, tarjoten monipuolisia mahdollisuuksia interaktiivisten näkymien luomiseen. Erityisesti Dash-kehys osoittautui tehokkaaksi työkaluksi, jonka avulla pystyttiin rakentamaan käyttäjäystävällinen sovellus Valioliigan pelaajatilastojen analysointiin.

Projektin toteutuksessa kohdattiin luonnollisesti myös haasteita. Aiheen rajaaminen ja aikataulutus olivat prosessin kriittisimpiä vaiheita. Jälkikäteen ajatellen suunnitteluvaiheeseen olisi kannattanut panostaa entistä enemmän, mikä olisi mahdollistanut vieläkin sujuvamman etenemisen.

Huolimatta pienistä vastoinkäymisistä, lopputulos oli onnistunut. Projekti ei pelkästään tuottanut toimivan sovelluksen, vaan myös syvensi ymmärrystä datan visualisoinnin mahdollisuuksista. Se herätti entistä suuremman kiinnostuksen tilastotieteisiin ja teknologian hyödyntämiseen tiedon analysoinnissa.

Tulevaisuuden näkymät ovat lupaavat. Datan visualisointi kehittyy jatkuvasti, ja tämä projekti toimi erinomaisena ponnahduslautana syvemmälle perehtymiselle. Jatkossa olisi mielenkiintoista tutkia vielä monimutkaisempia sovelluksia ja niiden käyttömahdollisuuksia eri liiketoimintaympäristöissä.

Tämä kokemus osoitti, että teknologia tarjoaa yhä enemmän työkaluja monimutkaisen tiedon ymmärtämiseen ja hyödyntämiseen, kunhan osaamme niitä oikein käyttää.

8 Yhteenveto

Opinnäytetyön tutkimuskysymykset olivat:

1. Miksi datan visualisointi on tärkeää?
2. Millä tavoin dataa voidaan visualisoida?
3. Mitä vaiheita datan visualisoimiseen liittyy?

Opinnäytetyön tutkimuskysymyksiin löytyi mielestäni vastaus. Työtä tehdessäni pääsin paremmin selville datan visualisoinnin eri vaiheisiin ja tutustumaan itselleni aiemmin tuntemattomaan Plotlyyn ja sen Dash-kehykseen, joka osoittautui mielekkääksi ja selkeäksi työkaluksi datan visualisointiin.

Opinnäytetyöprosessi oli itselleni hieman hankala sillä mieluisan aiheen löytäminen ei ollut helppoa. Aihe vaihtui alkumetreillä muutamaan kertaan, kunnes päädyin tähän ja tämänkin aiheen kanssa oli hankaluuksia mihin suuntaan opinnäytetyötä vien ja mitä teen käytännönosassa. Myöskään työelämän ja opinnäytetyön sekä sosiaalisen elämän yhtäaikaista yhdistämistä en kokenut helpoksi ja opinnäytetyön edistäminen oli hyvin satunnaista.

Alkuperäisessä aikataulussa en pysynyt mutta olen kuitenkin jokseenkin tyytyväinen, että sain valmiiksi kuitenkin saman vuoden puolella. Jälkiviisaana voisi sanoa, että opinnäytetyön aihetta ja itse opinnäytetyötä olisi voinut alkaa pohtimaan jo vähän ennen kuin vasta kun sitä pitäisi alkaa jo tekemään.

Opinnäytetyöprosessi on ollut opettavainen kokemus ja lisännyt kiinnostustani tilastotieteisiin ja datan visualisointiin. Se on lisännyt tietämystäni erilaisista datan visualisointityökaluista ja rajapinnoista.

Lähteet

Calzon, B. (2023). *Misleading Statistics Examples – Discover The Potential For Misuse of Statistics & Data In The Digital Age*. dataspine.com. https://www.datapine.com/blog/misleading-statistics-and-data/

Dash Bootstrap Components. (2024). *Dash Bootstrap Components*. https://dash-bootstrap-components.opensource.faculty.ai/

Dillemuth, J. (2022, joulukuuta 5). *Mihin Python-ohjelmointikieltä käytetään?* Tieturi. https://www.tieturi.fi/blogi/mihin-python-ohjelmointikielta-kaytetaan/

Hagberg, C. (2023, tammikuuta 19). *What is an ETL? Everything Marketers need to know*. https://funnel.io/blog/etl-explained

Han, J., & Kamber, M. (2006). *Data mining: Concepts and techniques* (2nd ed). Elsevier ; Morgan Kaufmann.

Ilyas, I. F., & Chu, X. (2019). *Data Cleaning* (1st edition). ACM Books.

Kilcommins, S. (2021, huhtikuuta 28). *Plotly Dash—Everything You Need To Know*. Medium. https://medium.datadriveninvestor.com/plotly-dash-everything-you-need-to-know-bc09a5e45395

Koponen, & Hilden. (2019). *Data Visualization Handbook*. Aalto University.

Majumder, P. (2021, lokakuuta 6). Guide to Create Interactive Plots with Plotly Python. *Analytics Vidhya*. https://www.analyticsvidhya.com/blog/2021/10/interactive-plots-in-python-with-plotly-a-complete-guide/

McKinney, W. (2017). *Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython* (2nd edition). O'Reilly Media.

Parikka, H., & Härkönen, T. (2020). *YRITYS- VASTUU ULOTTUU DATAAN*.

Plotly. (2024). *Dash HTML Components*. https://dash.plotly.com/dash-html-components

Plotly. (2024e). *Advanced Callbacks | Dash for Python Documentation | Plotly*. https://dash.plotly.com/advanced-callbacks

Plotly. (2024c). *Graph | Dash for Python Documentation | Plotly*. https://dash.plotly.com/dash-core-components/graph

Plotly. (2024b). *Part 1. Layout | Dash for Python Documentation | Plotly*. https://dash.plotly.com/layout

Plotly. (2024d). *Part 2. Basic Callbacks | Dash for Python Documentation | Plotly*. https://dash.plotly.com/basic-callbacks

Plotly. (2024a). *Plotly-express*. https://plotly.com/python/plotly-express/

Sukumar, H. (2024). *Heatmaps in Data Visualization: A Comprehensive Introduction*. Inforiver. https://inforiver.com/insights/heatmaps-in-data-visualization-a-comprehensive-introduction/

Underdahl, B. (2018). *Data Integration For Dummies®, 2nd Informatica Special Edition*.

Valcheva, S. (2017, joulukuuta 5). Types of Graphs and Charts and Their Uses: With Examples and Pics. *Blog For Data-Driven Business*. https://www.intellspot.com/types-graphs-charts/

VanderPlas, J. (2017). *Python Data Science Handbook: Essential Tools for Working with Data* (1st edition). O'Reilly Media.

Vij, R. (2022, heinäkuuta 22). *Understanding Statistical Data Types*. Medium. https://towardsdatascience.com/understanding-statistical-data-types-2993dafcac86

Özkan, I. (2023, elokuuta 22). *Data Protection Principles: The 7 Principles Of GDPR Explained*. https://www.cyberpilot.io/cyberpilot-blog/data-protection-principles-the-7-principles-of-gdpr-explained/

Liite 1: Aineistonhallintasuunnitelma

Aineisto tallennetaan ja sitä käsitellään opinnäytetyön tekijän omalla salasanalla suojatulla tietokoneella. Aineisto sijaitsee tietokoneen C-asemalla ja sisältää työssä tehdyn koodin ja analysoitavan csv-tiedoston.

Aineiston omistaa opinnäytetyön tekijä ja aineisto ei sisällä arkaluontoisia tietoja. Opinnäytetyön aineistoa ei jatkokäytetä ja opinnäytetyön tekijä säilyttää aineiston tietoturvallisesti vuoden ajan opinnäytetyön hyväksymispäivästä, jotta opinnäytetyön tulokset voidaan tarvittaessa varmistaa.