k-Nearest Neighbours  
Daniel van Eijk-Bos, Rick van Mourik

In de **main()** functie wordt als eerste de dataset ingelezen (samen met de bijbehorende dates en labels) en opgeslagen in variabelen voor later gebruik. Daarna kan je kiezen tussen het inladen van de dagen uit validatieset (samen met de bijbehorende dates en labels), of het inladen van de test data (zonder bekende dates en labels). De dates die eventueel ingelezen worden, worden rechtstreeks uit de data gehaald en de bijbehorende labels worden berekend in de **load\_data()** functie.  
Wanneer de data is ingelezen en opgeslagen, wordt deze genormaliseerd door middel van de **normalize()** functie. Deze functie gebruikt de minimale en maximale data die uit de originele data wordt gehaald met behulp van de **find\_min\_max()** functie. De **normalize()** functie gebruikt de **var – var\_min / var\_max – var\_min** formule om voor elk datapunt in een dag de genormaliseerde versie hiervan te berekenen en op te slaan.   
Afhankelijk van de wat je wilt doen (het algoritme testen met **validation1.csv** of seizoenen berekenen voor nieuwe dagen uit **days.csv**) gebeuren er nu verschillende dingen. (Ook is er de optie om de optimale k-value te berekenen, maar dit is onderdeel van het validatieproces) Voor het testen van nieuwe dagen, worden voor elke dag de **calculate\_distance()** en **pinpoint\_season()** functies aangeroepen. De **calculate\_distance()** functie berekent de afstand van de geselecteerde dag tot elk van de andere datapunten en geeft deze terug in een op afstand gesorteerde lijst gevuld met de afstanden tot deze datapunten en de corresponderende seizoenen. De **pinpoint\_season()** functie gaat de eerste k-aantal datapunten van deze list na, en telt op in welke seizoenen deze datapunten zich bevinden. Als er maar één seizoen is dat er het vaakst voorkomt, wordt deze terug gegeven. Als er meerdere seizoenen zijn die er het vaakst voorkomen, wordt de **pinpoint\_season()** functie recursief aangeroepen met **k – 1**. De uitkomst hiervan wordt daarna geprint.  
Bij de optie om data te valideren wordt de **success\_rate\_calculation()** functie aangeroepen. Deze doet in principe hetzelfde als wat hiervoor is uitgelegd, maar rekent dan ook met behulp van de daadwerkelijke antwoorden uit hoeveel % van onze berekeningen goed zijn.  
De laatste optie om de optimale k-waarde te berekenen, gebruikt hiervoor de **calculate\_optimal\_k()** functie. Deze functie berekent voor elke gewenste k-waarde wat het slagingspercentage is, en geeft het hoogste slagingspercentage terug samen met de corresponderende k-waarde.