



Hochschule **RheinMain**  
University of Applied Sciences  
Wiesbaden Rüsselsheim

# PRIVACY RANKING

## Wahlprojekt SS 2017

Letztes Update: 24. August 2017



Studiengang Informatik  
Hochschule RheinMain

# GLIEDERUNG

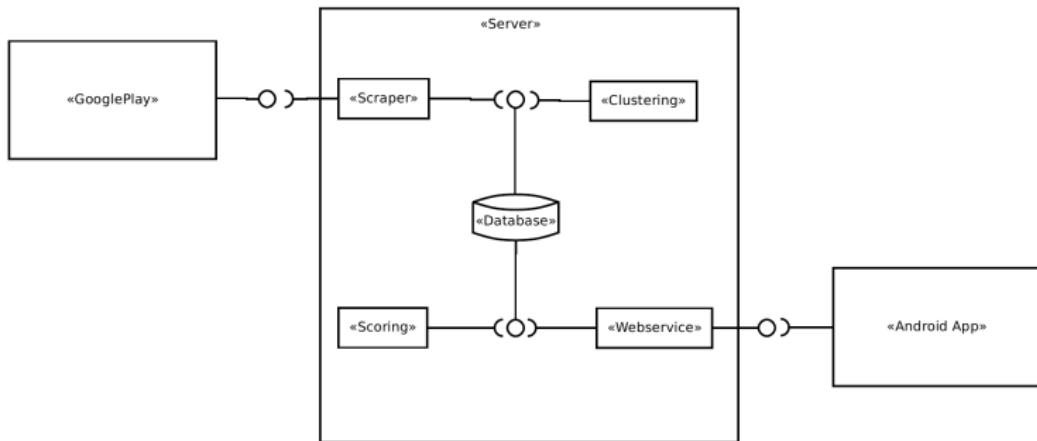
1. Einleitung
2. Die Anwendung
3. Webservice
4. Datenbeschaffung und Verarbeitung
5. Projektmanagement

# EINLEITUNG

# ANFORDERUNGEN AN DAS PROJEKT

- ▶ Kategorisierung und Bewertung der Berechtigungen
- ▶ Erstellung einer Applikation zur Darstellung der Ergebnisse

# ARCHITEKTUR



# LIVE-DEMO

# DIE ANWENDUNG

# WIE WURDE DIE APP ERSTELLT

- Android Studio ist eine freie Integrierte Entwicklungsumgebung (IDE)
- von Google entwickelt
- offizielle Entwicklungsumgebung für Android



# VORGEHENSWEISE

- ▶ Daten abfragen
- ▶ Daten verwalten
- ▶ Daten darstellen

# DATEN ABFRAGEN

JSON	Rohdaten	Kopfzeilen
Speichern	Kopieren	
<pre>App_id: "beleverion.com.torchlight" title: "Taschenlampe (alt)" title_en: "Flashlight (old)" developer_id: "Beleverion" ▶ description: "Never again wandering thou in the dark anymore!" ▼ short_description: "NEUE APP: market://details?id=com.belerion.torchlight" rating: "4" min_downloads: "5" max_downloads: "10" reviews: "74" cost: "0" currency: "USD" ▶ icon: "https://lh3.ggpht.com/Lu...KDHiiDpt8_9uU_A9qd-oVB0" app_category: "LIBRARIES_AND_DEMO" version: "1.3" min_android: "4.0.3" ▼ changelog: "Russische, Türkische und Englische Übersetzungen hinzugefügt." Permission_id: "11" Clustering_id: "0" cluster_id: "0" similarity: "76.7071"</pre>		
		10

# DATEN ABFRAGEN

Die Daten werden vom Server mit Hilfe der JSON geholt

```
{"App_id":"belerion.com.torchlight",
"title":"Taschenlampe (alt)",
"title_en":"Flashlight old)", ...}
```

```
String jsonStr = sh.makeServiceCall("http://privacyranking.cs.hs-rm.de/app/" + AppID);

if (jsonStr != null) {
    try {

        JSONObject c = new JSONObject(jsonStr);

        String title = c.getString("title");
        String title_en = c.getString("title_en");
        String App_id = c.getString("App_id");
        String rating = c.getString("rating");
```

# DATEN VERWALTEN

```
public class AppContact {  
  
    public String appId;  
    public String title;  
    public String title_en;  
    public String similarity;  
    public String permissions;  
    public String description;  
    public String rating;  
    public String min_downloads;  
    public String cost;  
    public String changelog;  
    public String link;  
    public String icon;  
    public boolean checkboxChecked;  
}
```

```
JSONObject c = new JSONObject(jsonStr);  
  
String title = c.getString("title");  
String title_en = c.getString("title_en");  
String App_id = c.getString("App_id");  
String rating = c.getString("rating");  
  
AppContact tempAppContact = new AppContact();  
tempAppContact.title = title;  
tempAppContact.title_en = title_en;  
tempAppContact.appId = App_id;  
tempAppContact.rating = rating;
```

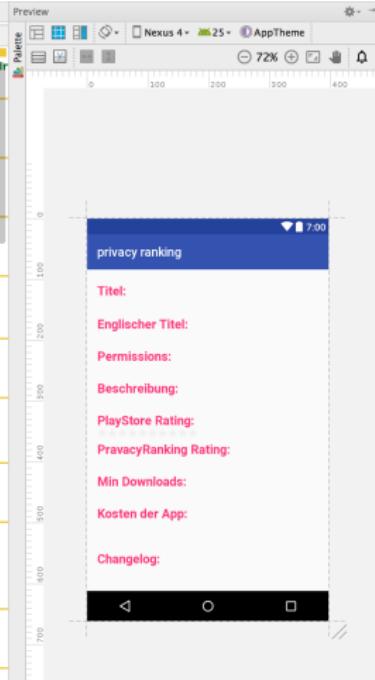
# DATEN DARSTELLEN - XML TEIL

The screenshot shows an Android Studio project structure on the left and the code editor for `list_item2.xml` on the right.

**Project Structure:**

- app**:
  - manifests**
  - java**:
    - `com.example.george.privacyranking`:
      - `About`
      - `AppCompContact`
      - `AppContact` (selected)
      - `Categorie`
      - `Categorie3`
      - `Categorie12`
      - `Categorie12Compare`
      - `CategorieArrayAdapter`
      - `CategorieArrayAdapterCompare`
      - `CategorieArrayAdapterInfo`
      - `CategorieArrayAdapterSearch`
      - `Compare`
      - `DownloadExampleActivity`
      - `DrawIcon`
      - `DrawView`
      - `HttpHandler`
      - `Search`
      - `Search2`
    - res**:
      - drawable**
      - layout**:
        - `about.xml`
        - `activity_main.xml`
        - `compare.xml`
        - `list_item.xml`
        - `list_item2.xml`
        - `list_item_cat.xml`
        - `list_item_cat_checkbox.xml`
        - `list_item_cat_search.xml`
        - `list_itemm.xml`
        - `search.xml`
      - menu**
      - mipmap**
      - values**
    - Gradle Scripts**

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:padding="16dp">
    <TextView
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:paddingTop="6dp"
        android:text="Titel"
        android:textColor="@color/colorAccent"
        android:textSize="20sp"
        android:textStyle="bold" />
    <TextView
        android:id="@+id/title"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:paddingBottom="2dp"
        android:textColor="@color/colorPrimaryDark"
        android:textSize="16sp"
        android:textStyle="bold" />
    <TextView
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:paddingTop="6dp"
        android:text="Englischer Titel"
        android:textColor="@color/colorAccent"
        android:textSize="20sp"
        android:textStyle="bold" />
    <TextView
        android:id="@+id/title_en"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:paddingBottom="2dp"
        android:textColor="@color/colorPrimaryDark"
        android:textSize="16sp"
        android:textStyle="bold" />
    <TextView
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:paddingTop="6dp"
        android:text="Permissions"
        android:textColor="@color/colorAccent"
        android:textSize="20sp"
        android:textStyle="bold" />
```



# DATEN DARSTELLEN

```
<TextView
    android:id="@+id/title"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:paddingBottom="2dip"
    android:textColor="@color/colorPrimaryDark"
    android:textSize="16sp"
    android:textStyle="bold" />

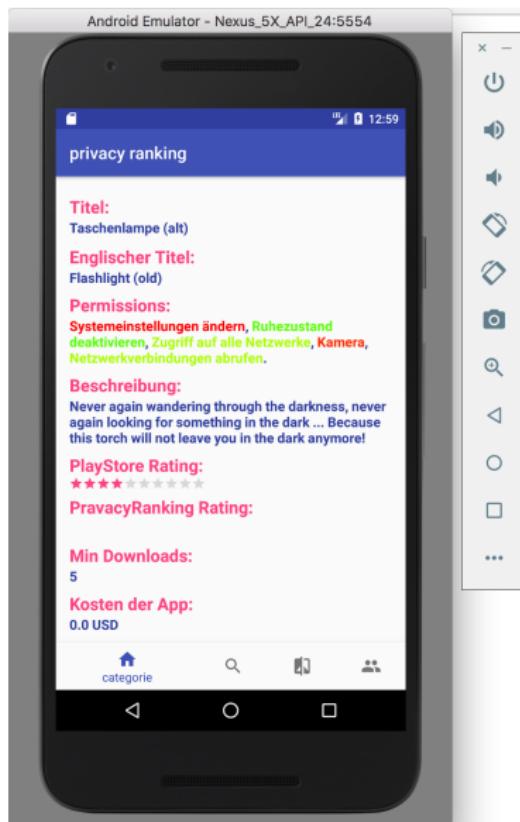
@Override
public View getView(int position, View convertView, ViewGroup parent) {
    AppContact appContact = getItem(position);

    if (convertView == null) {
        convertView = LayoutInflater.from(getContext()).inflate(R.layout.list_item2, parent, false);
    }

    TextView tvTitle = (TextView) convertView.findViewById(R.id.title);
    TextView tvTitle_en = (TextView) convertView.findViewById(R.id.title_en);

    tvTitle.setText(appContact.title);
    tvTitle_en.setText(appContact.title_en);
```

# ERGEBNIS



# WEBSERVICE

# WEBSERVICE

- ▶ Was ist ein Webservice?
- ▶ Warum wird er in diesem Projekt benötigt?
- ▶ Representational State Transfer (REST)

# SLIM FRAMEWORK

- ▶ Was ist Slim?
- ▶ Warum nicht <sup>„</sup>from scratch<sup>“</sup> selbst coden?
- ▶ Hat das auch Nachteile?

# DATENBANKVERBINDUNG

```
$config['db']['host']      = "localhost";
$config['db']['user']       = "XXXXXXXXXXXXXX";
$config['db']['pass']       = "XXXXXXXXXXXXXX";
$config['db']['dbname']     = "privacy_ranking";

$app = new \Slim\App(["settings" => $config]);
$container = $app->getContainer();
```

# DATENBANKVERBINDUNG

```
$container['db'] = function ($c) {
    $db = $c['settings']['db'];
    $pdo = new PDO("mysql:host=" . $db['host']
        . ";dbname=" .
        $db['dbname'] . ";charset=utf8",
        $db['user'], $db['pass']);
    $pdo->setAttribute(PDO::ATTR_ERRMODE,
        PDO::ERRMODE_EXCEPTION);
    $pdo->setAttribute(
        PDO::ATTR_DEFAULT_FETCH_MODE,
        PDO::FETCH_ASSOC);
    return $pdo;
};
```

## BEISPIEL ANFRAGE

```
$app->get('/perm/{id}', function ($request ,  
    $response , $args) {  
  
    try  
    {  
  
        $sth = $this->db->prepare("SELECT name ,  
            Permission_id , weight FROM Apps  
            NATURAL JOIN App_permissions NATURAL  
            JOIN Permissions WHERE App_id=:id");  
  
        $sth->bindParam("id", $args['id']);  
  
        $sth->execute();
```

# BEISPIEL ANFRAGE

```
$category = $sth->fetchAll();

if($category) {
    return $this->response->withJson(
        $category, 200);

} else {
    throw new PDOException('No
        Permissions needed.');
}

} catch(PDOException $e) {
    echo '[' . '{"name":' . $e->getMessage()
        . '}]' ;
}
});
```

# BEISPIEL ANFRAGE

- ▶ Anfrage an  
`http://privacyranking.cs.hs-rm.de/perm/com.tinder` wird gestellt.
- ▶ nginx leitet an Slim weiter
- ▶ Slim ruft `get('/perm/[id]')`... auf
- ▶ DB Anfrage wird vorbereitet und ausgeführt
- ▶ Ergebnis wird als JSON gepackt zurückgegeben

# BEISPIEL ANFRAGE ANTWORT

```
[{"name": "In-App-K\u00e4ufe", "Permission_id": "0",  
 "weight": "0.1"},  
 {"name": "Ge\u00e4te- &  
     App-Verlauf", "Permission_id": "1",  
 "weight": "0.7"},  
 {"name": "Standort", "Permission_id": "6",  
 "weight": "1"},  
 {"name": "Telefon", "Permission_id": "8",  
 "weight": "0.7"},  
 (...)]
```

# SWAGGER

The OpenAPI Specification (OAS)[formerly known as the Swagger Specification] defines a standard, language-agnostic interface to RESTful APIs which allows both humans and computers to discover and understand the capabilities of the service without access to source code, documentation, or through network traffic inspection. When properly defined, a consumer can understand and interact with the remote service with a minimal amount of implementation logic."

# SWAGGER

swagger bilder hinzufügen TODO

# DATENBESCHAFFUNG UND VERARBEITUNG

# WEBSITE GOOGLE PLAYSTORE

The screenshot shows the Pou app page on the Google Play Store. A modal window is open, listing permissions required by version 1.4.73:

- \$ In-App-Käufe
- Fotos/Medien/Dateien
  - USB-Speicherinhalte lesen
  - USB-Speicherinhalte ändern oder löschen
- Speicher
  - USB-Speicherinhalte lesen

Below the permissions, it says: "Bei Updates von Pou können in jeder Gruppe automatisch zusätzliche Funktionen hinzugefügt werden. [Weitere Informationen](#)".

At the bottom right of the modal is a "Schließen" button.

Below the modal, the developer tools Network tab is visible, showing network requests. The table has columns: Name, Status, Type, Initiator, Size, Time, Waterfall, and Duration. The requests listed are all 200 status code responses from "jquery.min.js:3".

Name	Status	Type	Initiator	Size	Time	Waterfall	Duration
getdoc?authuser=0	200	xhr	jQuery#AeW0sZageOM...	3.7 KB	73 ms		200.00 ms
data:image/png;base...	200	png	jquery.min.js:3	(from me...)	0 ms		
data:image/png;base...	200	png	jquery.min.js:3	(from me...)	0 ms		
data:image/png;base...	200	png	jquery.min.js:3	(from me...)	0 ms		
data:image/png;base...	200	png	jquery.min.js:3	(from me...)	0 ms		
data:image/png;base...	200	png	jquery.min.js:3	(from me...)	0 ms		
data:image/png;base...	200	png	jquery.min.js:3	(from me...)	0 ms		

9 requests | 3.8 KB transferred

# SCRAPING DER DATEN

- ▶ Zugriff auf den Webservice von Google
- ▶ <https://play.google.com/store/xhr/getdoc?authuser=0>
- ▶ POST (ids=app\_id, xhr=1)

```
[["gdar",1,[["me.pou.app","me.pou.app",1,3,"/store/apps/details?id\u003dme.pou.app","/store/apps/details?id\u003dme.pou.app","https://play.google.com/store/apps/details?id\u003dme.pou.app","https://market.android.com/details?id\u003dme.pou.app","Pou",...]
```

# EXTRAHIEREN DER DATEN

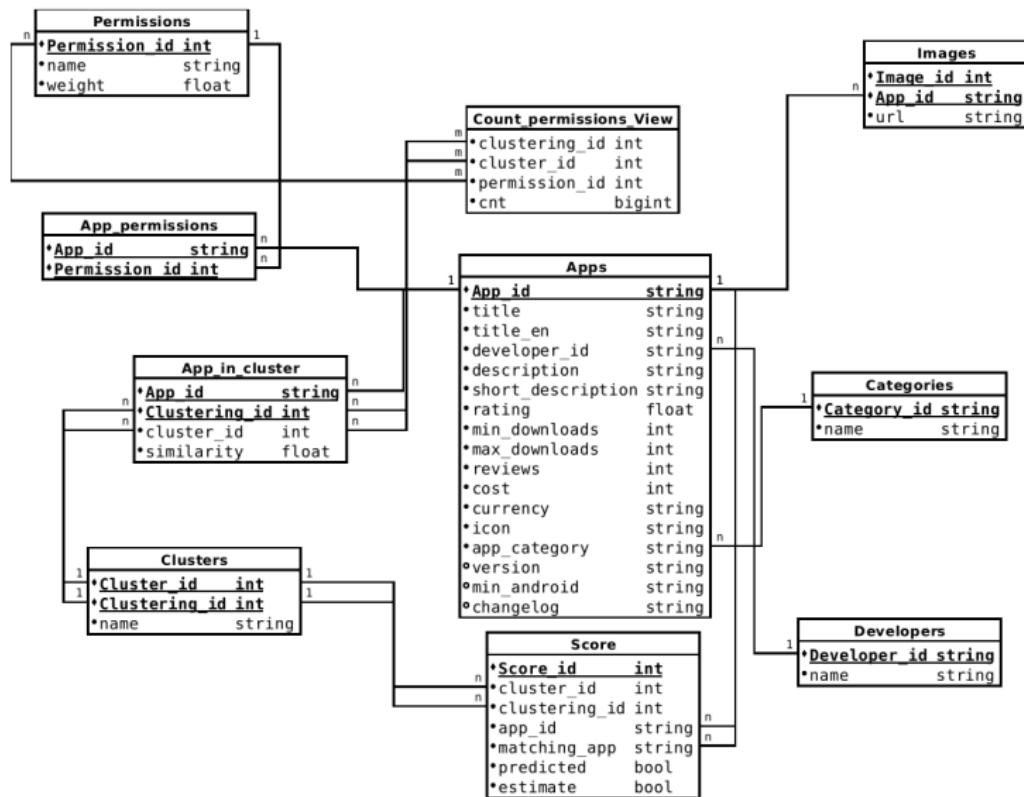
- ▶ Schreiben eines Wrappers in Python
- ▶ Lokalisieren der nötigen Informationen

```
def extract_title(data):
    return _remove_emojis(data[0][2][0][8])

def extract_description(data):
    return _remove_emojis(data[0][2][0][9])

def extract_rating(data):
    return data[0][2][0][23]
```

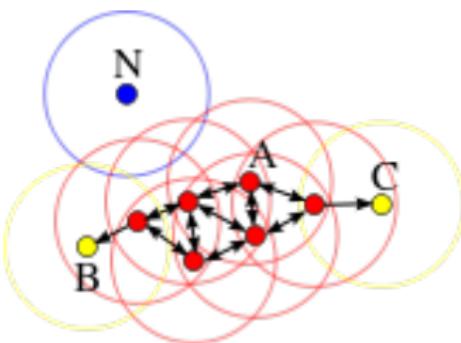
# MARIADB DATENBANK



# DATAMINING

- ▶ Kategorisierung mithilfe von Clustering
- ▶ Auswahl zwischen den einzelnen Algorithmen
  - ▶ K-Means
    - ▶ Anzahl Cluster muss bekannt sein
  - ▶ Affinity propagation
    - ▶ Terminiert nicht
  - ▶ Mean-Shift
    - ▶ Terminiert nicht
  - ▶ Ward hierarchical clustering
    - ▶ Terminiert nicht
  - ▶ DBSCAN
    - ▶ Rauschen

# DBSCAN



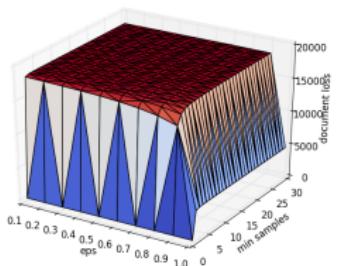
Quelle: Wikipedia

- ▶ Density-based spatial clustering of applications with noise
- ▶ Abstand (Epsilon) muss gut gewählt werden

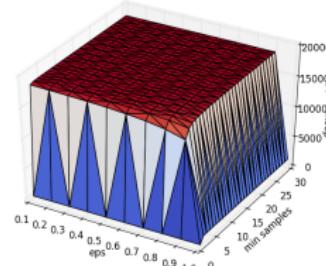
# TF-IDF

- ▶ Clustering-Algorithmen funktionieren nur mit numerischen Werten
- ▶ Text frequenzy
  - ▶ Je häufiger Wort in Text enthalten  $\Rightarrow$  bedeutend
  - ▶ Wert für *min-df* muss gut gewählt werden
- ▶ Inversed document frequenzy
  - ▶ Je häufiger Wort in allen Dokumenten enthalten  $\Rightarrow$  unbedeutend
  - ▶ Wert für *max-df* muss gut gewählt werden
- ▶ Dadurch entsteht Documents  $\times$  Features Matrix
- ▶ Max. Feautures werden bestimmt.

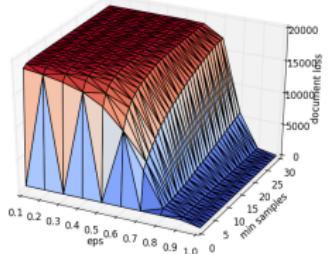
# GUTE METRIC FINDEN



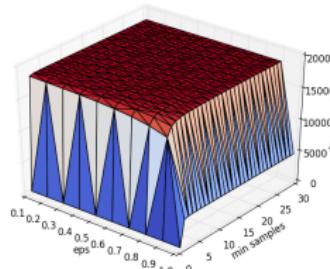
Euclidian



L2

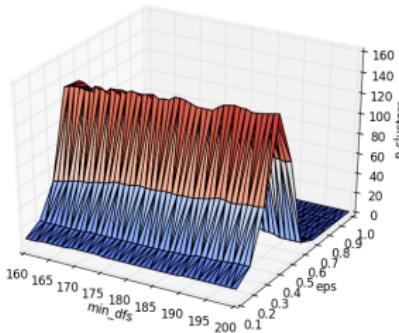
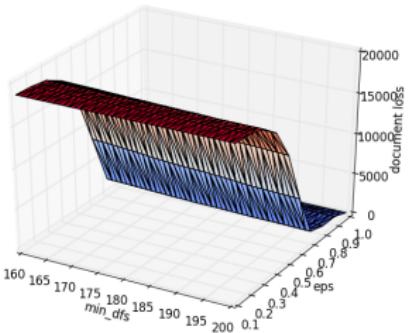
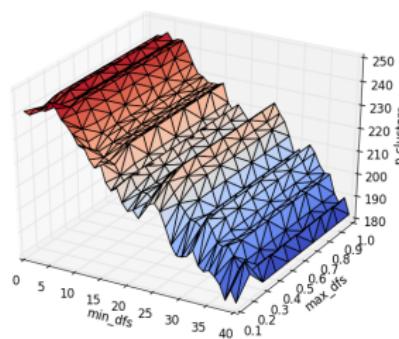
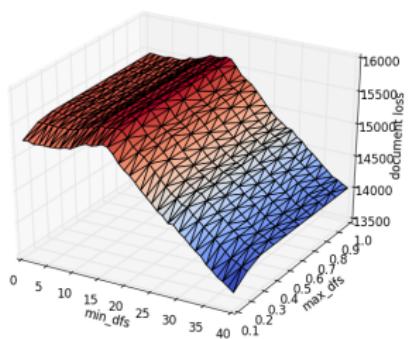


Cosine

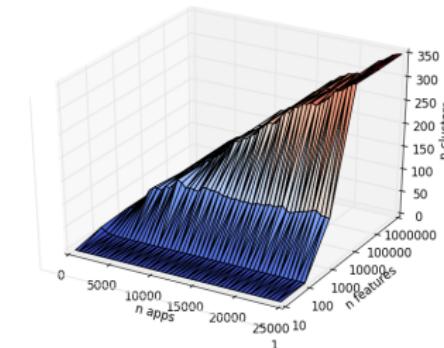
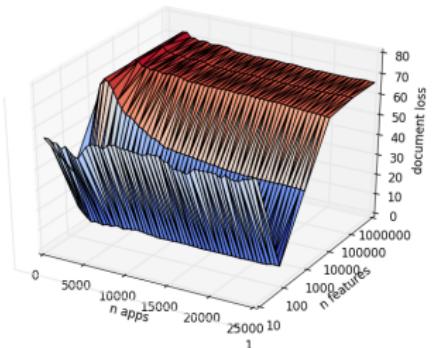
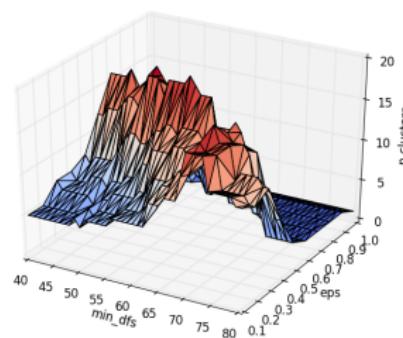
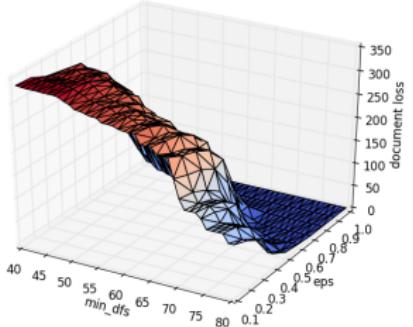


Minkowski

# GUTE PARAMETER FINDEN - TESTDATEN



# GUTE PARAMETER FINDEN - GOOGLE PLAY DATEN



# GUTE PARAMETER FINDEN

- ▶ max-df: 0.01
  - ▶ min-df: 0.005
  - ▶ eps: 0.45
  - ▶ min-samples: 30
  - ▶ features: 1500
- 
- ⇒ 42 Cluster
  - ⇒ Mehr als 50% Rauschen
  - ⇒ 1 Cluster viel zu groß

# KOMBINATION MIT ANDEREN ALGORITHMEN

- ▶ K-Means

- ▶ Anzahl Cluster aus DBSCAN → **mäßiger** Erfolg
- ▶ Anzahl GP Kategorien → **mäßiger** Erfolg

- ▶ Classifier

- ▶ DecisionTree → **miserabler** Erfolg
- ▶ BernoulliNB → **miserabler** Erfolg
- ▶ MLP → **miserabler** Erfolg
- ▶ AdaBoost → **miserabler** Erfolg
- ▶ KNeighbors → **akzeptabler** Erfolg

⇒ Kein Verlust mehr durch Rauschen

⇒ Zu großer Cluster wurde noch größer

⇒ Cluster beinhaltet mehr als 50% Apps

# HIERARCHICAL DBSCAN

Aufteilung von zu großen Clustern in kleinere.

⇒ Sprengt den Arbeitsspeicher.

*Dies liegt an der mieserablen Implementierung in SKLearn. Es ist besser, wenn du's selbst implementierst.  
- Viele Leute bei Stackoverflow*

Eigene Variante in Kombination mit KNeighbors:

- ▶ Zu große Cluster werden erneut mit DBSCAN geclustert (kleineres Epsilon)
  - ▶ Dabei entstandenes Rauschen wird mithilfe KNeighbors neu verteilt
- ⇒ Clusterqualität wurde schlechter, kein guter Erfolg

# BEWERTUNG DER APPS

Die Apps werden nach dem Einfluss auf die Privatsphäre bewertet.

1. Sammeln der Berechtigungen innerhalb eines Clusters

Permissions

0	4	9	10	11	12
---	---	---	----	----	----

Mit den Berechtigungen:

ID	Name
0	In-App-Purchases
4	Calender
9	Pictures/Media/Files
10	Storage
11	Camera
12	Microphone

# BEWERTUNG DER APPS

## 2. Berechnung der Gewichtung

Besteht aus zwei Teilen:

- ▶ Relative Häufigkeit von Apps die diese Berechtigung **nicht** haben

0.4	0.8	0.6	0.2	0.0	0.6
-----	-----	-----	-----	-----	-----

- ▶ Bosheit der Berechtigungen

0.1	0.6	0.1	0.1	0.9	0.9
-----	-----	-----	-----	-----	-----

# BEWERTUNG DER APPS

Diese werden miteinander multipliziert.

Permissions

0	4	9	10	11	12
0.04	0.48	0.06	0.02	0.0	0.54

### 3. Füllen der Matrix

Permissions

Apps	ID	0	4	9	10	11	12
	14	0.04	0.0	0.0	0.02	0.0	0.54
	42	0.0	0.48	0.06	0.0	0.0	0.0
	145	0.04	0.0	0.0	0.02	0.0	0.0
	465	0.04	0.0	0.06	0.02	0.0	0.54
	1010	0.0	0.0	0.0	0.02	0.0	0.0

# BEWERTUNG DER APPS

## 4. Aufsummieren der Werte

Apps	Permissions							$\Sigma$
	0	4	9	10	11	12		
14	0.04	0.0	0.0	0.02	0.0	0.54		<b>0.6</b>
42	0.0	0.48	0.06	0.0	0.0	0.0		<b>0.54</b>
145	0.04	0.0	0.0	0.02	0.0	0.0		<b>0.06</b>
465	0.04	0.0	0.06	0.02	0.0	0.54		<b>0.66</b>
1010	0.0	0.0	0.0	0.02	0.0	0.0		<b>0.02</b>

# BEWERTUNG DER APPS

## 5. Aufteilen in 3 Gruppen mithilfe K-Means

Apps	ID	$\sum$
	14	0.6
	42	0.54
	145	0.06
	465	0.66
	1010	0.02

- ▶ Gut - Grün
  - ▶ 80 - 120 degree
- ▶ Mittel - Gelb
  - ▶ 30 - 79 degree
- ▶ Schlecht - Rot
  - ▶ 0 - 29 degree

```
# 0 - 100
value = 100 - ((app_values[i] - min_value) *
    100.0) / (max_value - min_value)
# min_range - max_range
value = (value * (color_range[1] -
    color_range[0])) / 100) + color_range[0]
```

# PROJEKTMANAGEMENT

# PROJEKTMANAGEMENT

## Aufteilung der Arbeit

- ▶ Projektleiter: George
- ▶ Tech-Support: Rodion
- ▶ App-Erstellung: George, Viktor und Rodion
- ▶ Webservice: Simon
- ▶ Data-Mining, GoogleScraper, Scoring, Datenbank: Robert

## Zeitmanagement

- ▶ Teamtreffen jede Woche montags um 9:30 Uhr
- ▶ Treffen mit Herrn Igler mittwochs um 10:00 Uhr
- ▶ Meilensteine wurden festgelegt

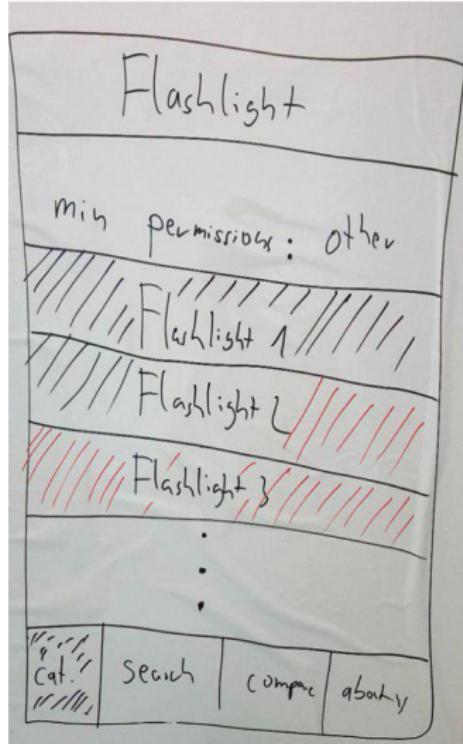
# PROJEKTMANAGEMENT

## Meilensteine

- ▶ 10.05.2017 Vorstellung Grob-Entwurf und Einigung auf Projektziel
- ▶ 24.05.2017 Grundgerüst der App
- ▶ 21.06.2017 Erste funktionierende Version
- ▶ 25.08.2017 App ist final und voll funktionsfähig

# PROJEKTMANAGEMENT

## Meilensteine



privacy ranking	
PERMISSIONS	MIN/MAX
FLASHLIGHT	
FLASHLIGHT X	
FLASH	
CAT4	
CAT5	
CAT6	
CAT7	
CAT8	
CAT9	
CAT10	

categorie

# PROJEKTMANAGEMENT

## Meilensteine

privacy ranking	
أحوال المتنفس	<input type="checkbox"/>
1 permission(s)	
Restaurant Innenarchitekturen	<input type="checkbox"/>
2 permission(s)	
LED Digital clock LWP	<input type="checkbox"/>
2 permission(s)	
Bluetooth-GPS-Lizenz	<input type="checkbox"/>
1 permission(s)	
LED Control Pro [ROOT]	<input type="checkbox"/>
1 permission(s)	
Fancy LED digital clock LWP	<input type="checkbox"/>
2 permission(s)	

categorie

privacy ranking	
TF: Screen-Beleuchtung Klassik	
0 permission(s)	
GO Keyboard Remove Ads	
0 permission(s)	
Kairo XP (for HD Widgets)	
0 permission(s)	
MDK Battery	
0 permission(s)	
Xposed Additions Pro	
0 permission(s)	
TF: Klassische LED Leuchte	
0 permission(s)	

categorie

# PROJEKTMANAGEMENT

## Kommunikation und Dokumentation

- ▶ Telegramm (Kommunikation)
- ▶ Slack(Jibble) (Zeiterfassung der Arbeitszeit)
- ▶ Wiki (Dokumentation des Projektes)
- ▶ Github (Repository mit all unseren Daten )

# FAZIT

## Probleme im Projekt

- ▶ Mussten anfangs mit Dummy Daten arbeiten
- ▶ Daten und Webservice standen am Anfang noch nicht zur Verfügung
- ▶ Clustering war noch nicht optimiert und hat zu große Cluster generiert
- ▶ Clustering- und Scoring-Algorithmus musste angepasst werden
- ▶ Cluster wurden verkleinert, leider mit Qualitätseinbußen

# FAZIT

## Haben wir das Projekt erfolgreich umgesetzt?

- ▶ Daten abgerufen ✓
- ▶ Daten gespeichert ✓
- ▶ Daten geclustert ✓
- ▶ Daten bewertet ✓
- ▶ Verbindungsmöglichkeit von App zu Daten ✓
- ▶ Nutzbares Endprodukt ✓

# FAZIT

## Was hätten wir besser machen können?

- ▶ App-Design
- ▶ Performance von der App
- ▶ Leicht um neue Datensätze erweiterbar
- ▶ Nicht nur auf GooglePlay Store beschränkt (Bspw. F-Droid)

ENDE

# Fragen?

Link zur App:

