Spotify Analysis

(Red) Shift Happens
Kelsey Li, Gloria Majchrzak, Stevan Thomas

Agenda:

Introduction / Exploration

Descriptive Analytics

Predictive Analytics

GitHub Link to View

Spotify Dataset

Original:

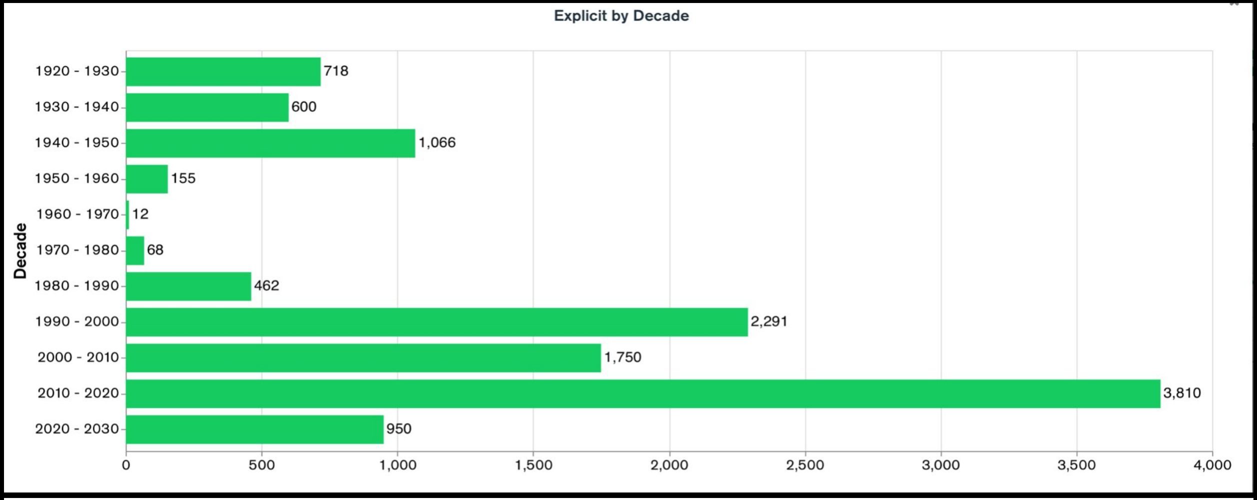
- Audio features of tracks
- Track count: 586,672 tracks
- Track features (variables): 20
- Time frame: 1922-2021

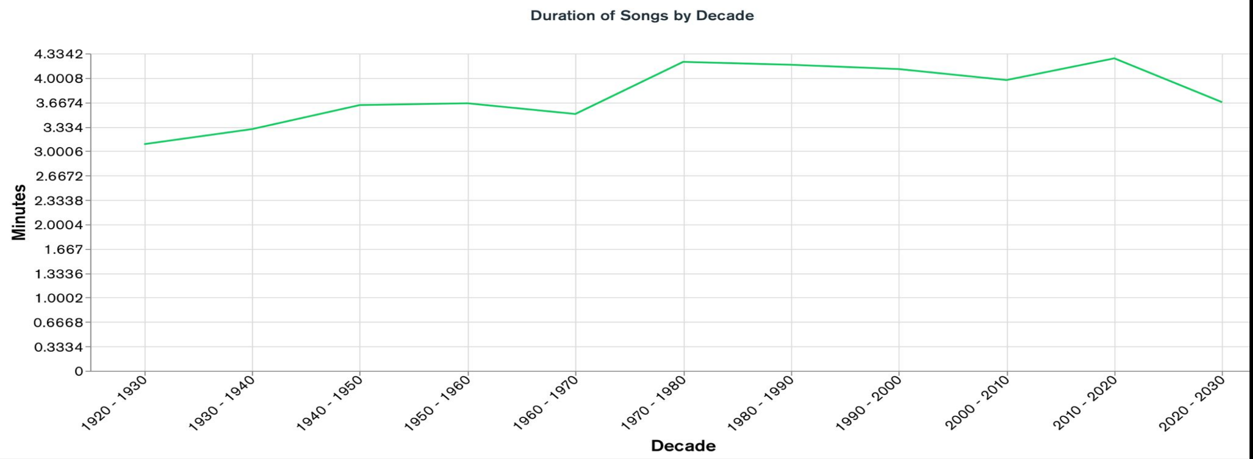
Prepared:

- Decade: created from release_date
- Artist1, Artist2, Artist3, Artist4: created by splitting list in artist variable
- Is_popular: binary created from cutoff of popularity variable
- N_artists: count of artists for each track

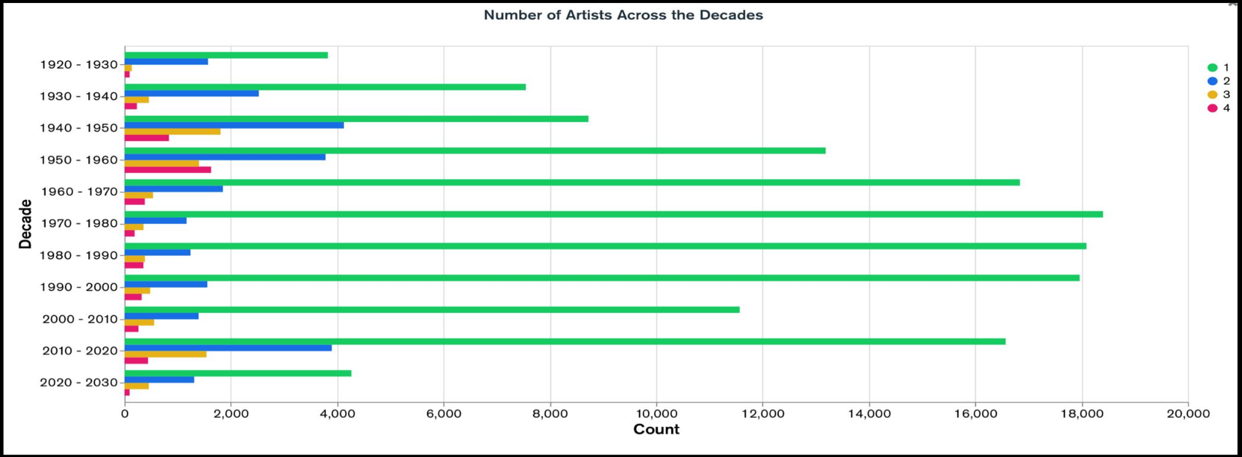
Data Type	**Features**
Primary	id (generated by Spotify)
Numeric	Acousticness, danceability, energy, duration_ms, instrumentalness, valence, popularity, tempo, liveness, loudness, speechiness
Dummy (0,1)	Mode, explicit
Categorical	Key, time signature, artists, id_artists, release_date, name

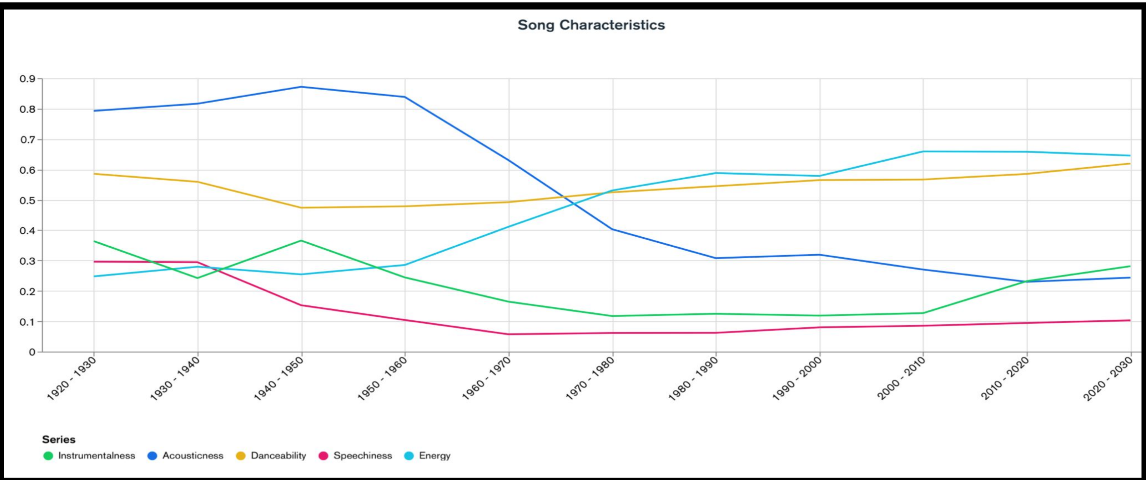
Descriptive Analytics


Explicit by Decade


Duration of Songs by Decade

Descriptive Analytics cont.





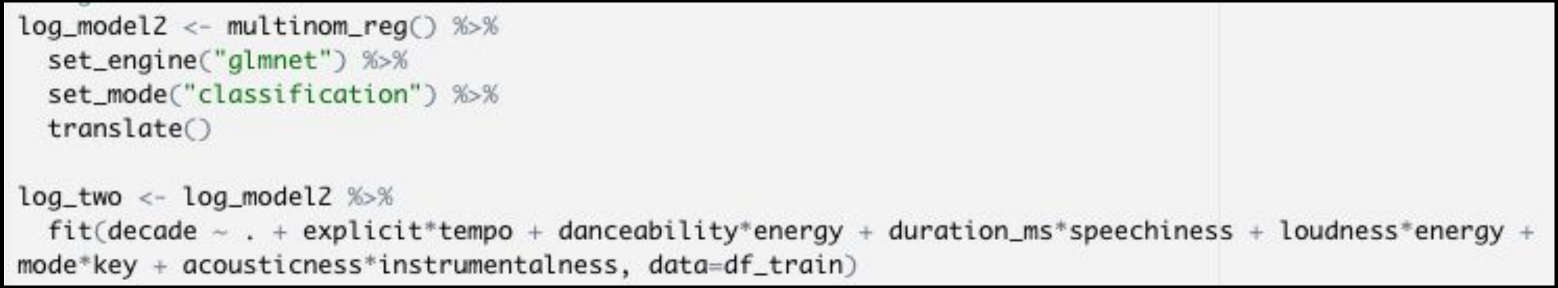
Predictive Analytics -
Multinomial Logistic Regression

Model 1:

- Factorize decade variable
- 15 numeric variables
- Data split: 75-25
- Accuracy: 0.3106

Model 2: Model 1 + Interaction

- Factorize decade variable
- 15 numeric variables
- 6 interaction terms
- Accuracy: 0.3157

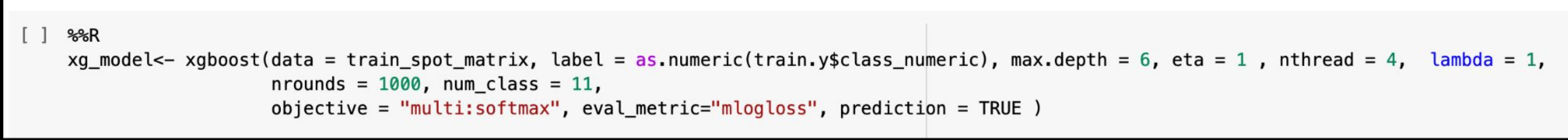


```
log_model2 <- multinom_reg() %>%
  set_engine("glmnet") %>%
  set_mode("classification") %>%
  translate()

log_two <- log_model2 %>%
  fit(decade ~ . + explicit*tempo + danceability*energy + duration_ms*speechiness + loudness*energy +
mode*key + acousticness*instrumentalness, data=df_train)
```

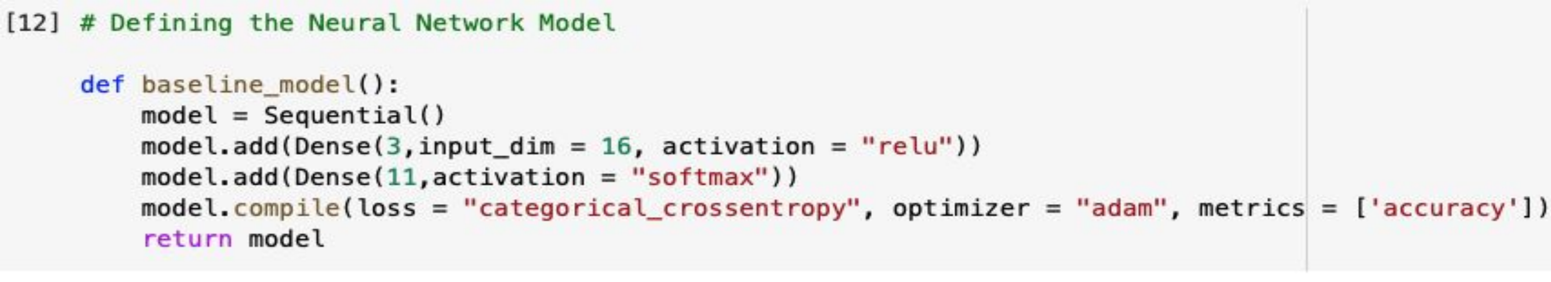
Academic paper: Kwak C, Clayton-Matthews A. Multinomial logistic regression. Nurs Res. 2002 Nov-Dec;51(6):404-10. doi: 10.1097/00006199-200211000-00009. PMID: 12464761.

Model 3: **XGBoost**



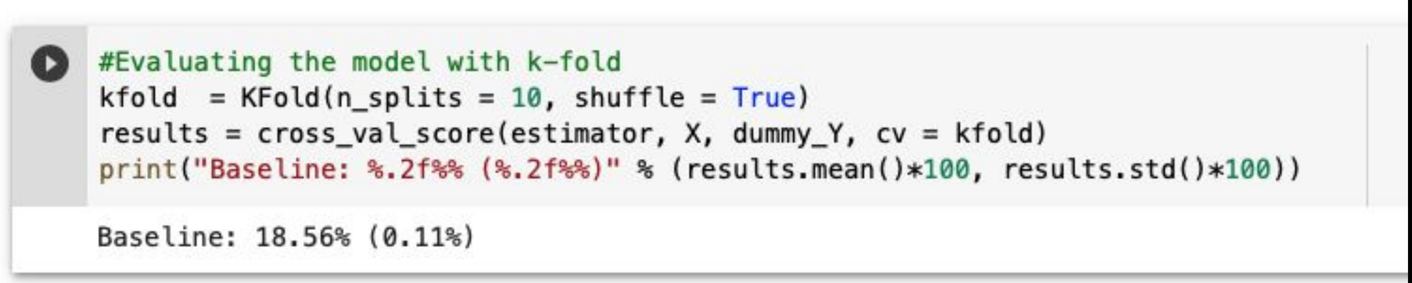
```
[ ]  %%R
     xg_model<- xgboost(data = train_spot_matrix, label = as.numeric(train.y$class_numeric), max.depth = 6, eta = 1 , nthread = 4,  lambda = 1,
                            nrounds = 1000, num_class = 11,
                            objective = "multi:softmax", eval_metric="mlogloss", prediction = TRUE )
```

Model 4: **Neural Networks**



```
[12] # Defining the Neural Network Model

     def baseline_model():
         model = Sequential()
         model.add(Dense(3,input_dim = 16, activation = "relu"))
         model.add(Dense(11,activation = "softmax"))
         model.compile(loss = "categorical_crossentropy", optimizer = "adam", metrics = ['accuracy'])
         return model
```



```
#Evaluating the model with k-fold
kfold  = KFold(n_splits = 10, shuffle = True)
results = cross_val_score(estimator, X, dummy_Y, cv = kfold)
print("Baseline: %.2f%% (%.2f%%)" % (results.mean()*100, results.std()*100))
```

```
Baseline: 18.56% (0.11%)
```