Human Gender Classification Using Machine Learning

TAGHREED ALAMRI

Overview

- Gender Classification problem
- Dataset
- EDA
- Feature Engineering
- Models (KNN, SVM and LR)
- Tools
- Conclusion and Result

Gender Classification

- Gender classification is to determine a person's gender, e.g., male or female.
- Face is one of the most important biometric traits .
- The goal of this project is to classify whether a person is male or female considering the facial features (such as nose width, Forehead length, etc.) of that person.
- I applied three classification models, k-nearest neighbors (KNN), Logistic Regression and Support Vector Machine (SVM)

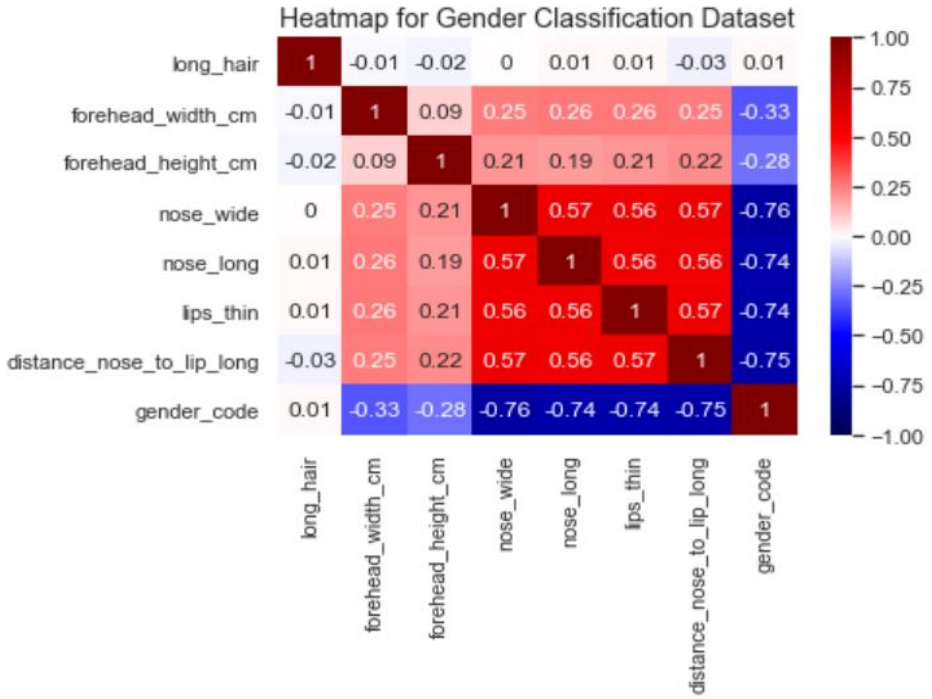
Dataset

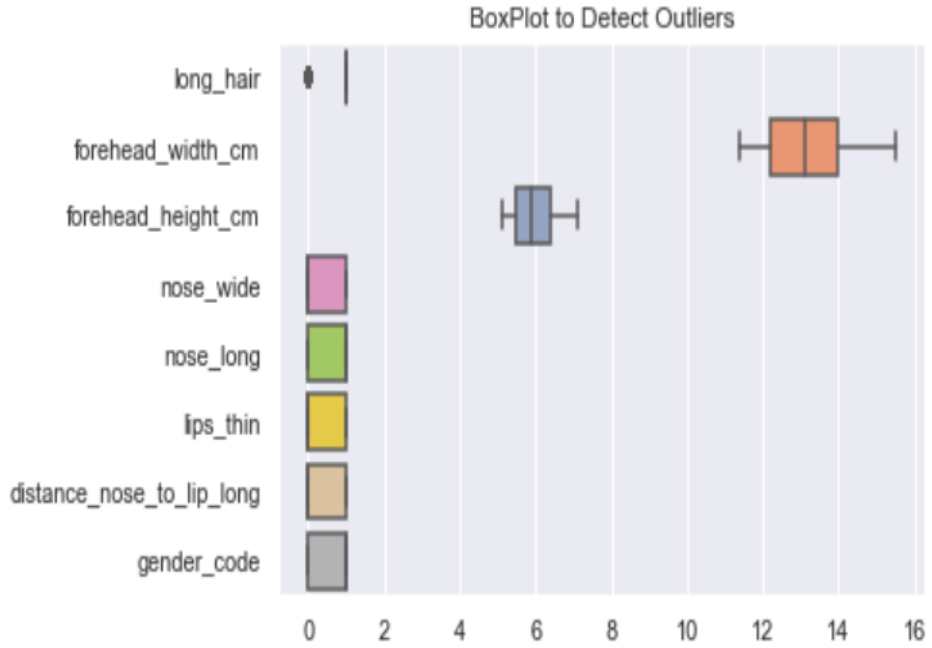
- The data I used is the Gender Classification dataset from Kaggle. It has 5001 samples that consists of 8 columns (7 features/predictors and 1 label/target column)

long_hair	Indicates whether this person has a long hair (1) or not (0).
forehead_width_cm	Width of the forehead from right to left given in cm.
forehead_height_cm	Height of the forehead in cm from where the hair grows to the eyebrows.
nose_wide	Whether the nose is wide or not. 1 represents wide and 0 not.
nose_long	Whether the nose is long or not. 1 represents long and 0 not.
lips_thin	Whether this person has a thin lip or not. 1 represents thin and 0 not.
distance_nose_to_lip_long	Is the distance from nose to lip is long? 1 represents yes and 0 not.
Gender	Either Meal or Female

- It contains 2501 Male samples and 2500 Female samples, so the dataset is balanced.

EDA


Heatmap for Gender Classification Dataset
long_hair 1 -0.01 -0.02 0 0.01 0.01 -0.03 0.01
forehead_width_cm -0.01 1 0.09 0.25 0.26 0.26 0.25 -0.33
forehead_height_cm -0.02 0.09 1 0.21 0.19 0.21 0.22 -0.28
nose_wide 0 0.25 0.21 1 0.57 0.56 0.57 -0.76
nose_long 0.01 0.26 0.19 0.57 1 0.56 0.56 -0.74
lips_thin 0.01 0.26 0.21 0.56 0.56 1 0.57 -0.74
distance_nose_to_lip_long -0.03 0.25 0.22 0.57 0.56 0.57 1 -0.75
gender_code 0.01 -0.33 -0.28 -0.76 -0.74 -0.74 -0.75 1
long_hair
forehead_width_cm
forehead_height_cm
nose_wide
nose_long
lips_thin
distance_nose_to_lip_long
gender_code
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00


BoxPlot to Detect Outliers
long_hair
forehead_width_cm
forehead_height_cm
nose_wide
nose_long
lips_thin
distance_nose_to_lip_long
gender_code
0
2
4
6
8
10
12
14
16

Feature Engineering

- Gender was converted from categorical to numerical with male= 0 and Female= 1 in a new column called "gender_code".
- Dataset records were split into 60/40 train vs. test(holdout).
- StandardScaler() was used to scale the features and found it gave better accuracy for KNN and Logistic Regression (For SVM I didn’t apply it because I found that it decreases the accuracy).

K-Nearest Neighbors Model(KNN

- I used GridSearchCV to Tune Hyperparameters of my KNN model with cv=5 and paramgrid shown in table:

'n_neighbors'	[1 – 100]
weights	['uniform', 'distance']

- Best parameters are {'n_neighbors': 94, 'weights': 'uniform’}

Support Vector Machine Model(SVM)

- I used GridSearchCV to Tune Hyperparameters of my SVM model with cv=5 and paramgrid shown in table:

Kernel	['rbf', 'sigmoid', 'linear']
C	[0.1, 1, 10, 100, 1000]
gamma	[1, 0.1, 0.01, 0.001, 0.0001]

- Best parameters are {‘kernel': 'rbf’, ‘C': 10, 'gamma': 0.1}

Logistic Regression Model(LR)

- I used GridSearchCV to Tune Hyperparameters of my LR model with cv=5
  And paramgrid shown in table:

C	[0.1, 1, 10, 100]
penalty	['l1', 'l2']
solver	‘liblinear’

- Best parameters are {'C': 10, 'penalty': 'l1’}

Tools

The main tools I used in addition to Python and Jupyter Notebook are:

- Numpy and Pandas for data manipulation.
- Scikit-learn for modeling.
- Matplotlib and Seaborn for plotting and visualization.

Conclusion

- KNN gave best performance on testing data with 97.15% accuracy. Then, SVM with 96.95% and the worst one is LR with 96.45% accuracy.

Metric	*KNN*	*SVM*	*LR*
Accuracy	*97.15%*	*96.95%*	*96.45%*

- GridSearchCV take more computational time to tune the parameters so to improve the computational time, we can try RandomizedsearchCV.