

ProJECT REPORT

Increment 3



**Zeshan Ather, John Goza, Hongkun Jin, Mouqing Jin**

Team 3

# **Abstract**

This project contributes to the current state of research by leveraging both shallow and deep learning techniques on video data to create a time based linear score prediction, as well as demonstrating the effectiveness of multi factor prediction with software modeling. We got very good accuracy while training the deep learning models e.g. 94% for CNN

# **Introduction**

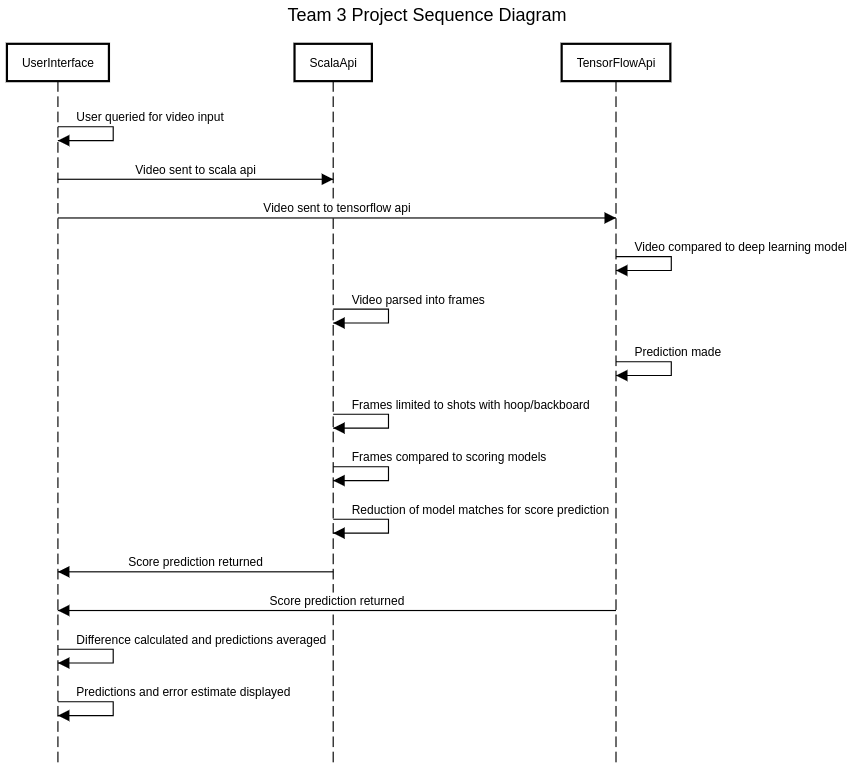
By leveraging competing software models against each other we believe we can significantly enhance correct software prediction of the desired output, the score of the basketball game. we can summarize the long basketball video into short highlight video and that will attract the more audience and will be able to get more views ratings. Our aim is also to predict some analytics for audience about winning team, goal ratios etc.

# **Related work**

In the field of audio-based methods [1] Baillie M, Jose JM (2003) Audio-based event detection for sports video. In: CIVR, pp 300–309 URL:<https://pdfs.semanticscholar.org/bb30/8c656104a833e8edc88219953211277a2a7a.pdf> [2] Xiong Z, Radhakrishnan R, Divakaran A, Huang TS (2003) Audio events detection based highlights extraction from baseball, golf and soccer games in a unified framework. In: IEEE ICASSP, pp 632–635 URL:<http://ieeexplore.ieee.org/document/1200049/> [3] Xu M, Maddage NC, Xu C, Kankanhalli MS, Tian Q (2003) Creating audio keywords for event detection in soccer video. In: IEEE ICME, pp 281–284 URL:<http://ieeexplore.ieee.org/document/1221608/> 2 In the field of motion-based methods [4] Ekin A, Tekalp AM, Mehrotra R (2003) Automatic soccer video analysis and summarization. IEEE Trans Image Process 12(7):796–807 URL:<http://ieeexplore.ieee.org/document/1212655/> [5] Pan H, van Beek PJL, Sezan MI (2001) Detection of slow-motion replay segments in sports video for highlights generation. In: IEEE ICASSP, pp 1649–1652 URL:<http://ieeexplore.ieee.org/document/941253/> 3 In the field of object and people detection-based methods [6] Hsu C, Chen H, Chou C, Ho C, Lee S (2014) Trajectory based jump pattern recognition in broadcast volleyball videos. In: ACM MM, pp 1117–1120 URL:<https://dl.acm.org/citation.cfm?id=2654985> 4 In the field of search-based methods [7] Chu L, Jiang S, Wang S, Zhang Y, Huang Q (2013) Robust spatial consistency graph model for partial duplicate image retrieval. IEEE Trans Multimedia 15(8):1982–1996 URL:<http://ieeexplore.ieee.org/document/6544623/> [8] Sun M, Farhadi A, Seitz S (2014) Ranking domain-specific highlights by analyzing edited videos. In: ECCV, pp 787–802 URL:<https://pdfs.semanticscholar.org/5c7a/dde982efb24c3786fa2d1f65f40a64e2afbf.pdf> [9] Yang H, Wang B, Lin S, Wipf D, Guo M, Guo B (2015) Unsupervised extraction of video highlights via robust recurrent auto-encoders. In: IEEE ICCV, pp 4633–4641 URL:<https://www.cv-foundation.org/openaccess/content_iccv_2015/papers/Yang_Unsupervised_Extraction_of_ICCV_2015_paper.pdf> 5 This paper authors’ previous intelligent basketball arena work [10] Liu W, Liu J, Gu X, Liu K, Dai X, Ma H (2017) Deep learning based intelligent basketball arena with energy image. In: MMM, pp 601–613 URL: <https://link.springer.com/chapter/10.1007/978-3-319-51811-4_49>

# **Approach**

This project will attempt a threefold approach to accurately model the score of any provided basketball footage. First, a TensorFlow deep learning model will be trained on a dataset with prominent scoreboard and scoring play footage to create an untrained model. Secondly, all footage will be compared to supervised, trained models created in Spark and tensorflow. Spark models will focus on shot types, scoreboard processing, and distinct motion for different scoring models



## **Data Sources**

Our data source is different high deficiency basketball videos. We collected some videos and then generated frames from each video. We also collected some long videos trim that video to extract specific data frames for the training of our model





## **Data Specification:** Amount, Format, Training/Testing data

We are using 4 categories of the data which are dunks, free throws, tapping and shooting. Each category contains 50 main frame images.

For Shallow Learningwe are splitting our data into 0.7 and 0.3 for training and testing respectively while training the naïve Bayes, k-Means, Decision tree and random forest. Input data is simply images

For Deep Learningwe are first converting .png image dataset into MNIST data format which are gray scale images. The common parameters for the model are

1. MNIST validation size is 40
2. Epochs: 500



# **Implementation details**

## **Analytic Algorithms/Platform**

## **Clustering**

In shallow learning we use K-Means clustering. The cluster size is 400 and number of iterations we used are 20

## **Regression/Classification**

**In TensorFlow linear regression** is applied. We used MNIST converted format dataset. The paraments for model are. Learning rate 0.01 Batch size 28 Epoch 500. We applied **softmax regression** model with same dataset format

### **Linear Regression**

Parameters setting:



## **Deep Learning**

**In TensorFlow** we applied the convoluted neural network model. For convoluted neural network model, we used 3 layers of filtering of 32,64 and 1024. We used the Adam optimizer for our CNN model.

### **SoftMax**

Parameters setting: None.

There are two .py files to build model and test. We combine them to calculate the time the running cost.

### **Results**

1. Running Time: 2.1413497924804688 seconds

### **CNN**

**Parameters setting**:

weight variable and bias variable are 0.1 and 0.1

The Optimizer we use is: AdamOptimizer(1e-4)

Filter size is: 32-64-1024

**Changing epochs**

We can see the results; the training accuracy always is 1 after step100. So, we changed epoch to 100 and see what happen.

## **Shallow Learning Model**

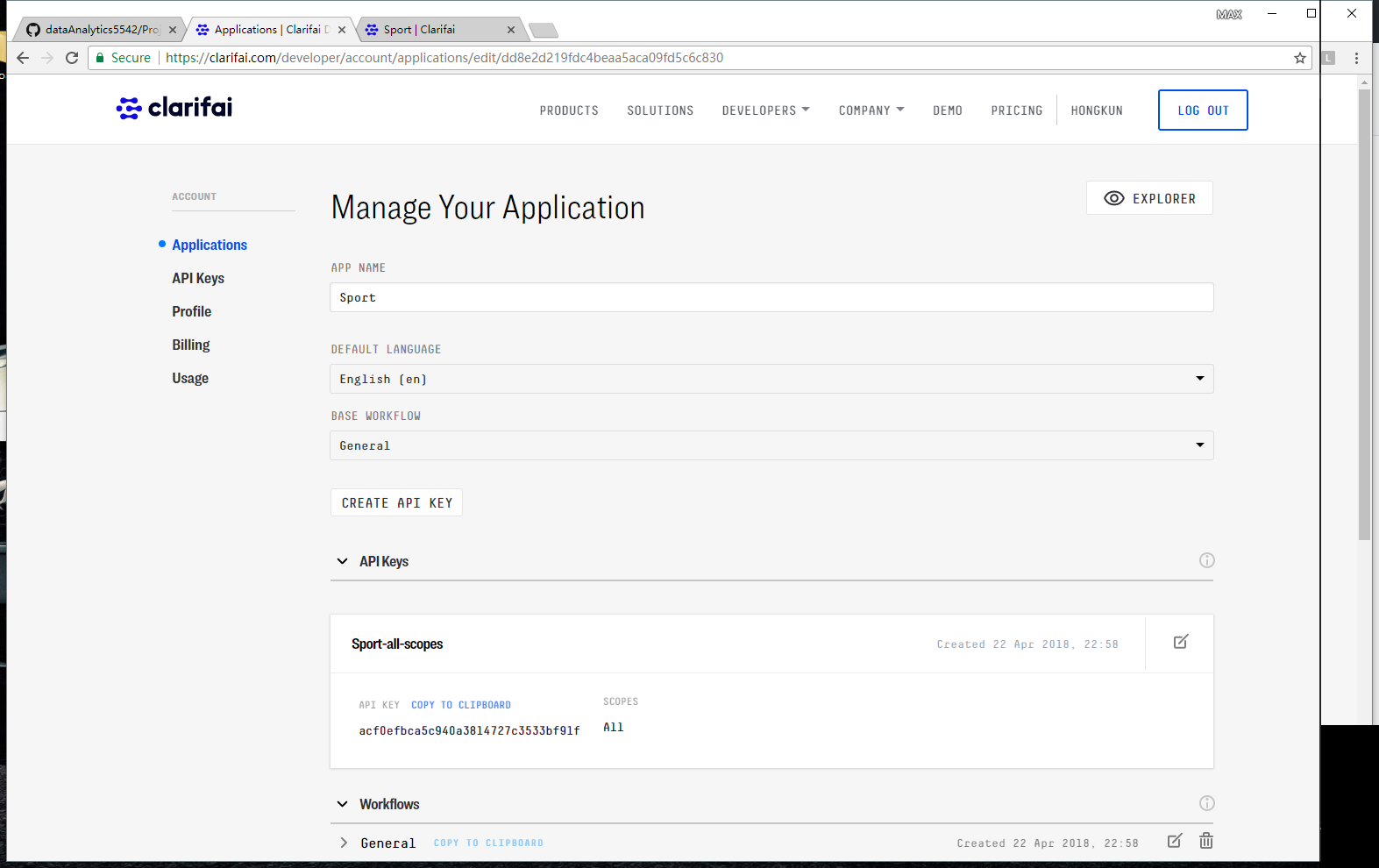
In shallow learning we applied **Naïve base, random forest and decision trees**. For training and testing purposes we split our data in to 70% and 30% ratio respectively.

## **Annotation (Clarifai API):**

Custom concepts to be trained: Dribbling, Dunk, Free-throws, Shooting

### **Step 1: Create an Application and Explore the UI**

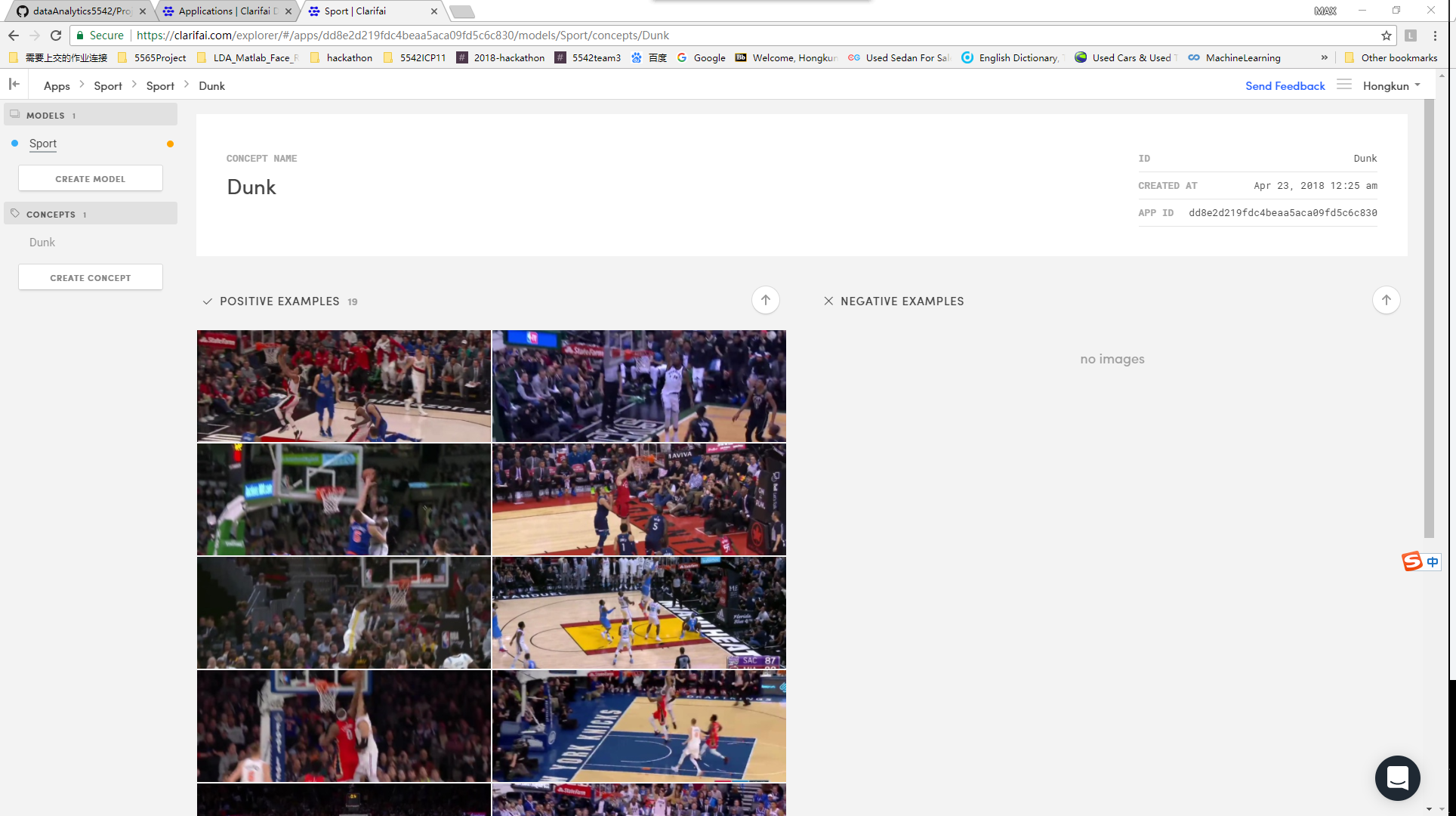
API calls (operations) are tied to an account and application.

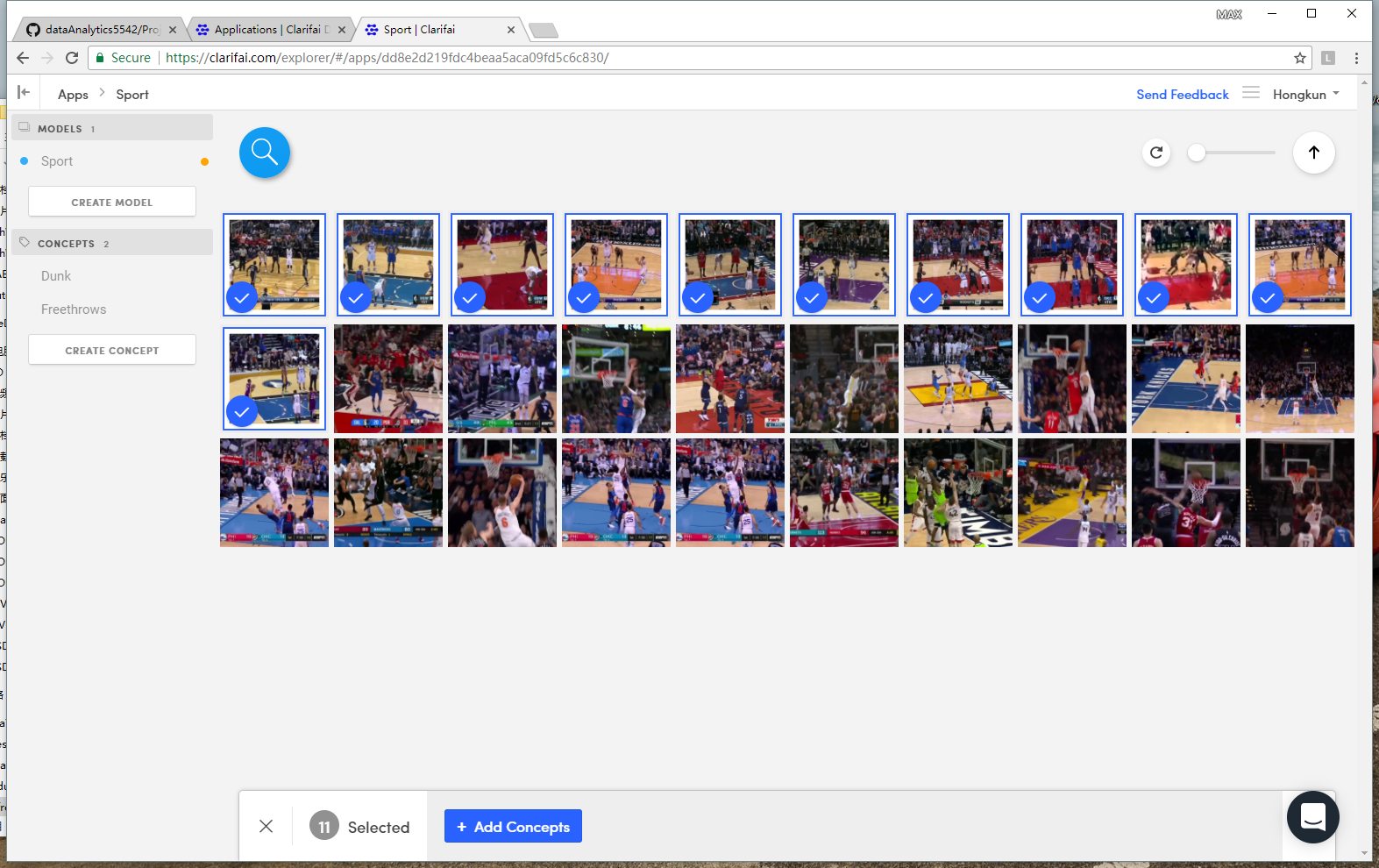


### **Step 2: Add images to application**

Custom models are built by training on our own data. The model will be able to make predictions specific to my own unique content and context.

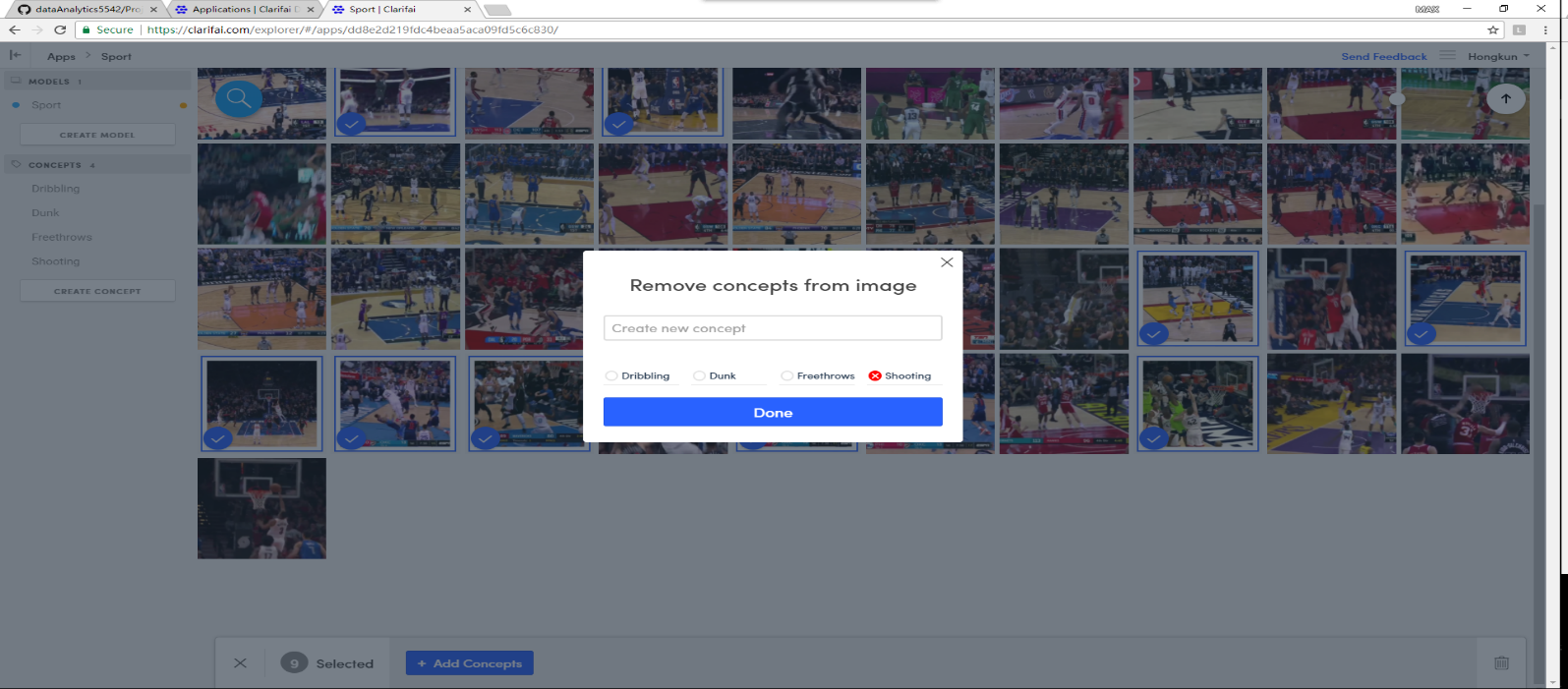
### **Step 3: Adding the first concept to those images**



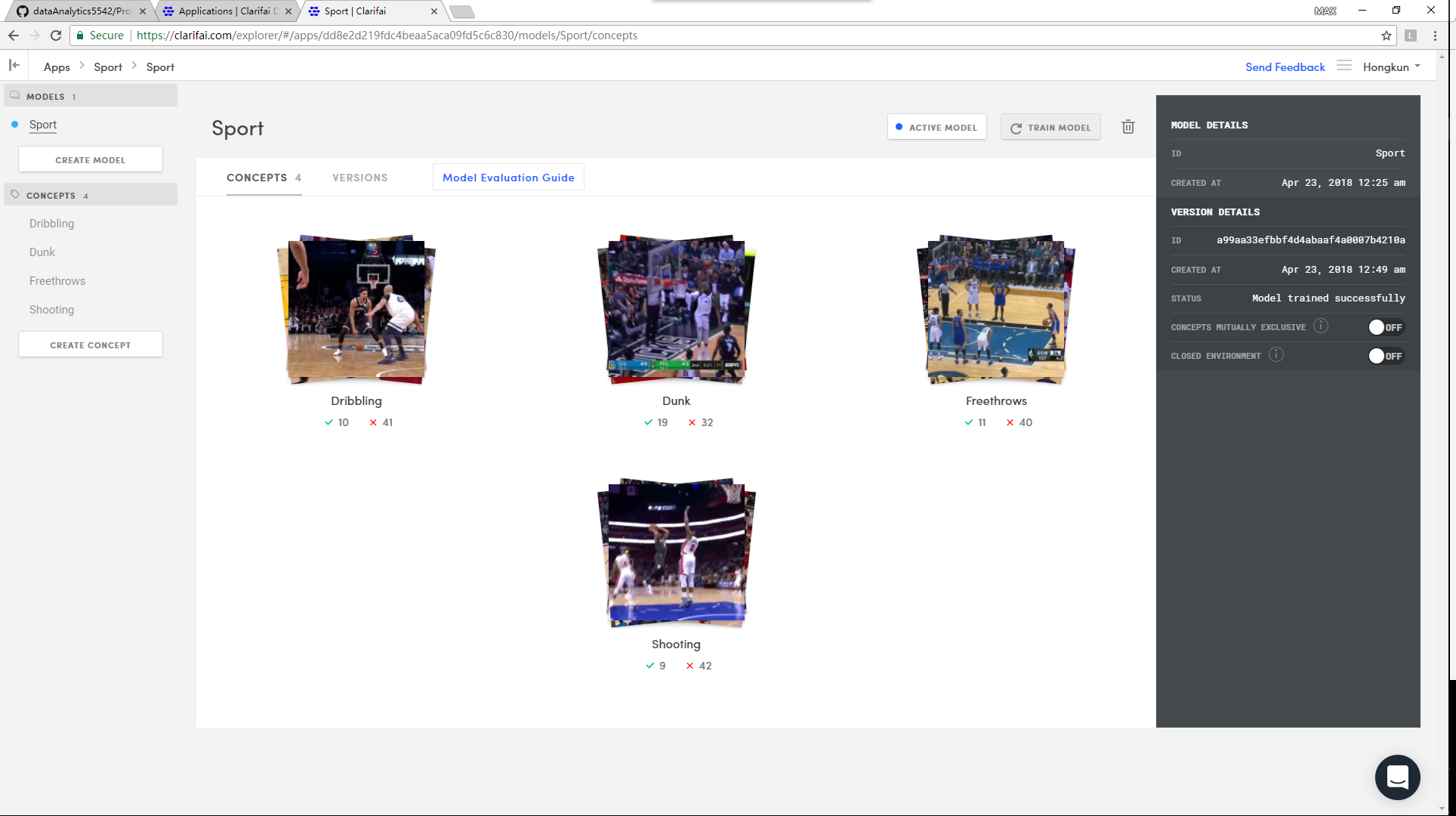


### **Step 4: Provide negative examples for that concept ('jet engine')**

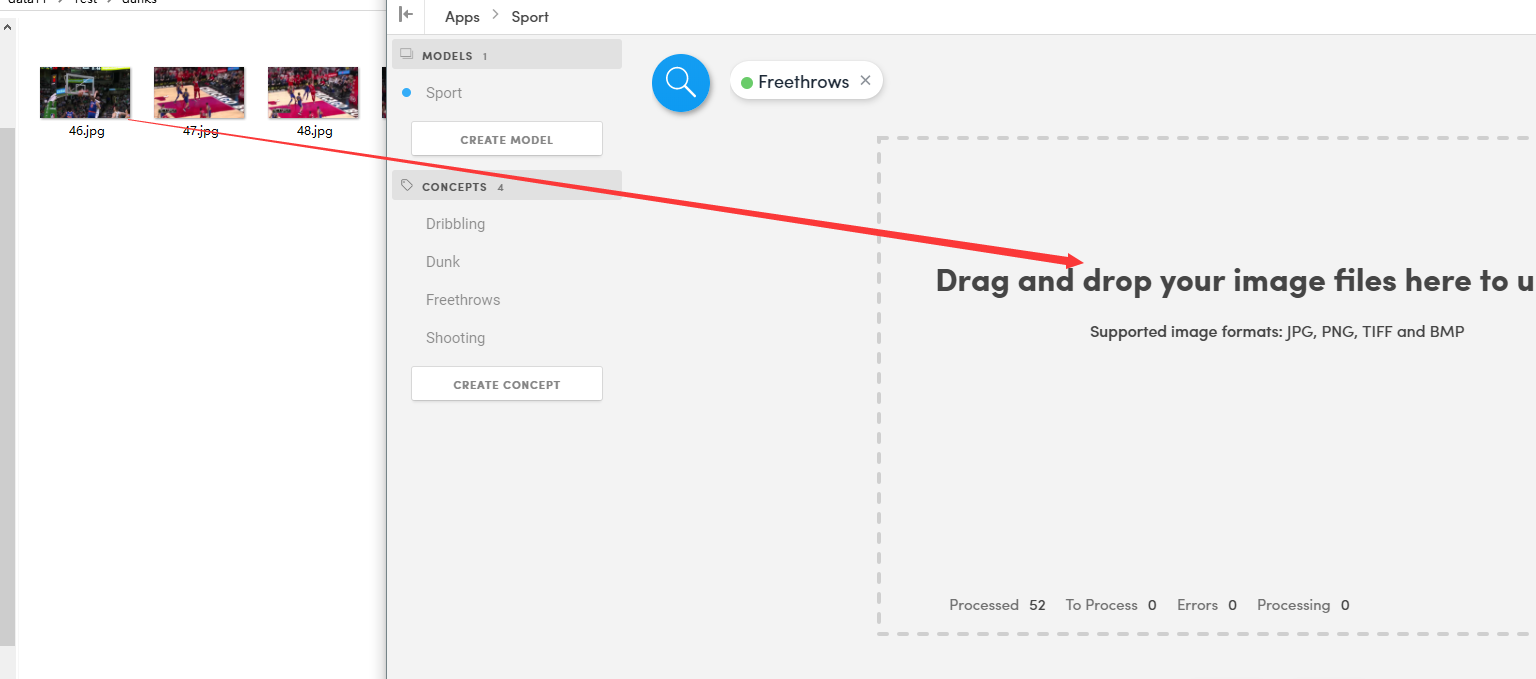
A robust and well performing concept is made up of both positive and negative examples.

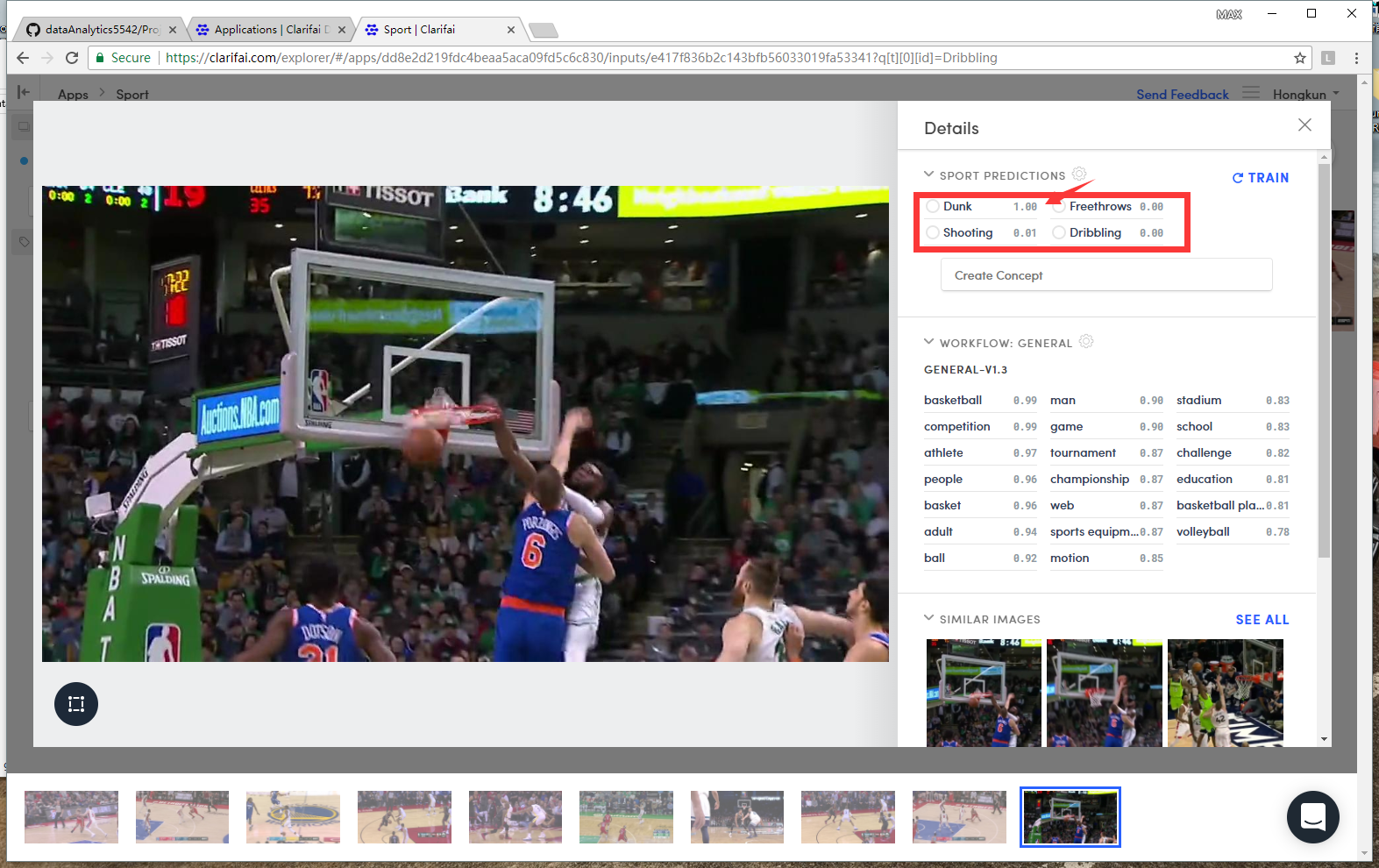


### **Step 5: Adding additional concepts to the model**



### **Step 6: Observe the model predictions and updates in real-time**





## **Training classifier with different feature**

# **Results**

|  |  |
| --- | --- |
| **Shallow Learning** | |
| Naïve Bayes | 33% |
| Decision Tree | 89% |
| Random Forest | 93% |
| **TensorFlow** | |
| CNN | 94% |
| Linear Regression | 67% |
| SoftMax Regression | 89% |

## **Clarifai API**

This is the model performance after custom training

|  |  |
| --- | --- |
| **Categories** | **Accuracy** |
| Dunk | 0.98 |
| Dribbling | 0.93 |
| Shooting | 0 |
| Free throws | 0.906 |
| Avg. Accuracy | 0.706 |

1. Running Time: 33.87053728103638 seconds

## **CNN:**

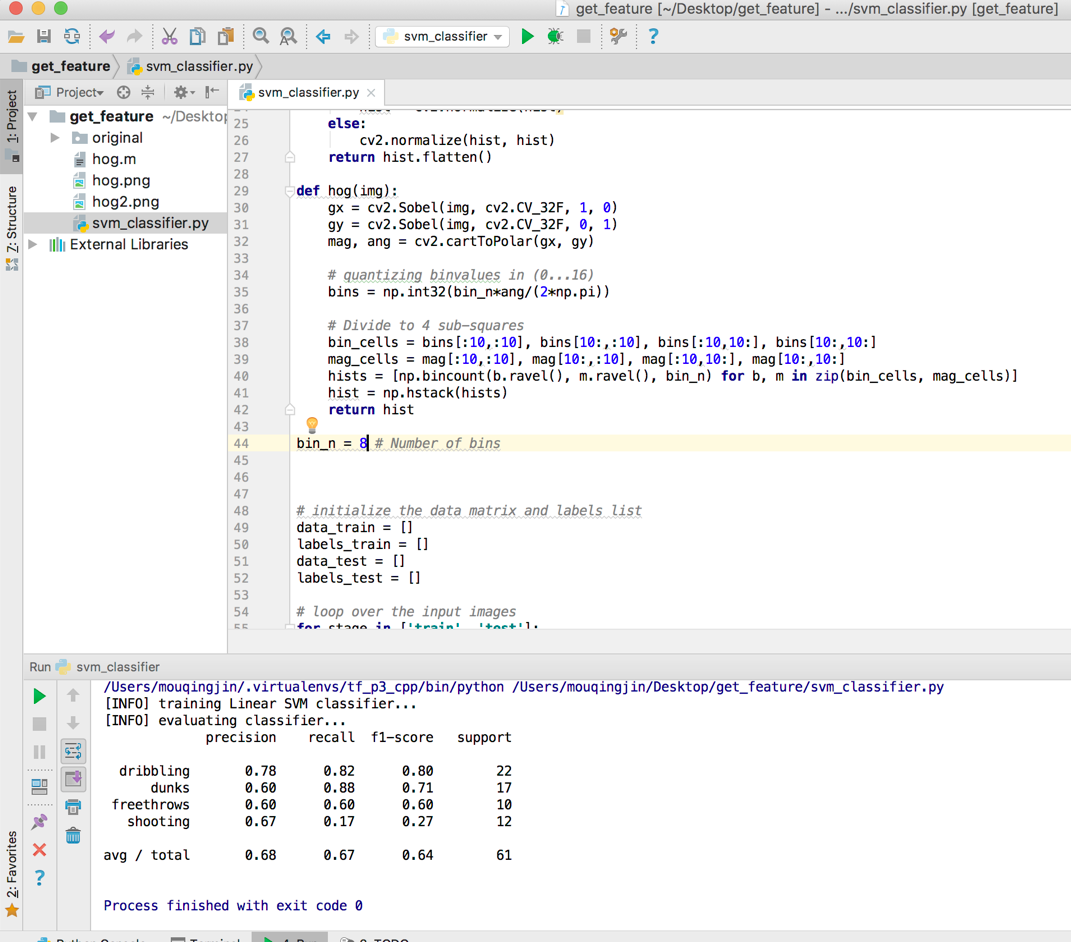
1. Running Time: 64.0976209640503 seconds

### **Changing Epoch**

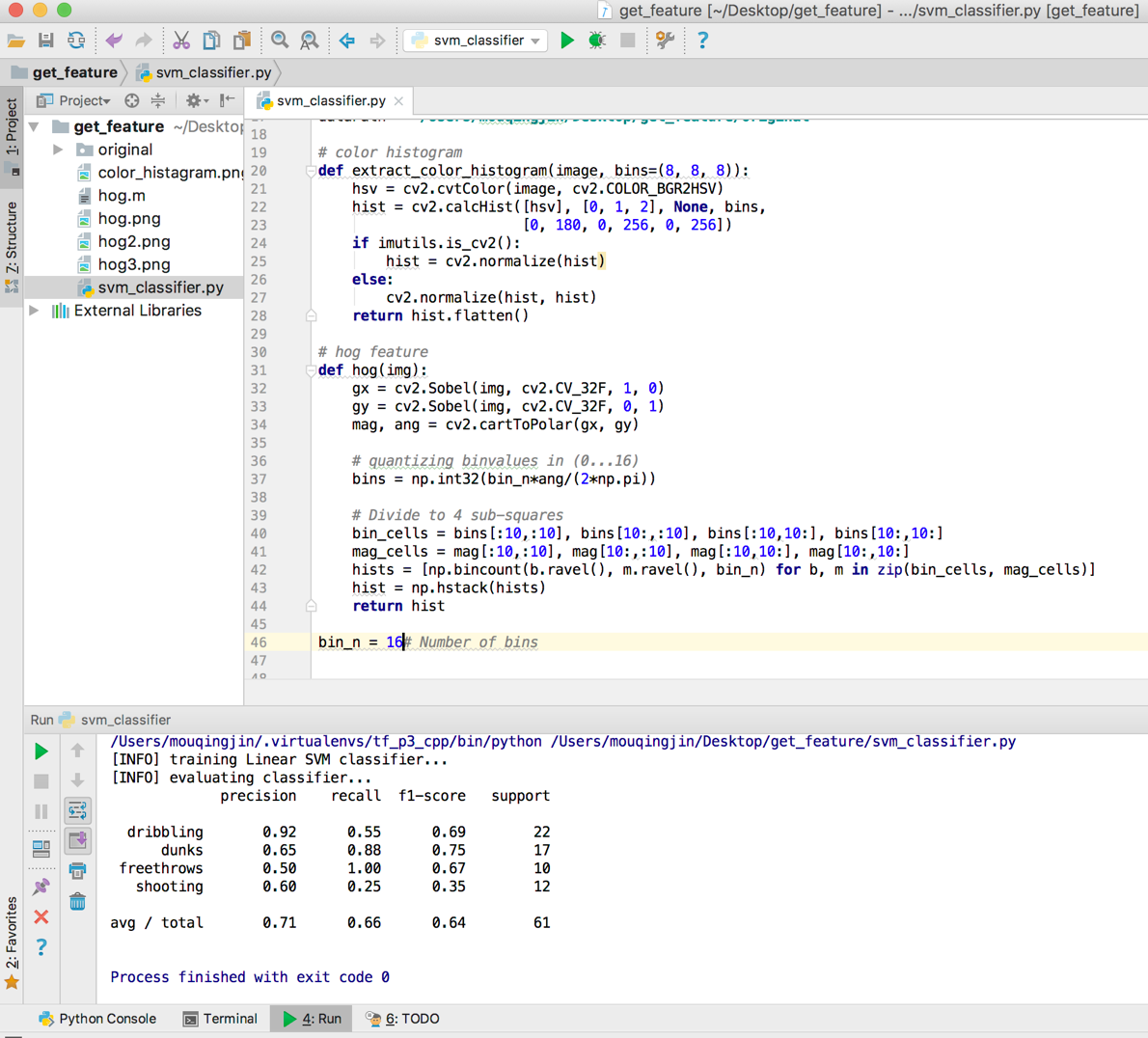
1. Running Time: 15.982191562652588 seconds

## **Training SVM with different feature**

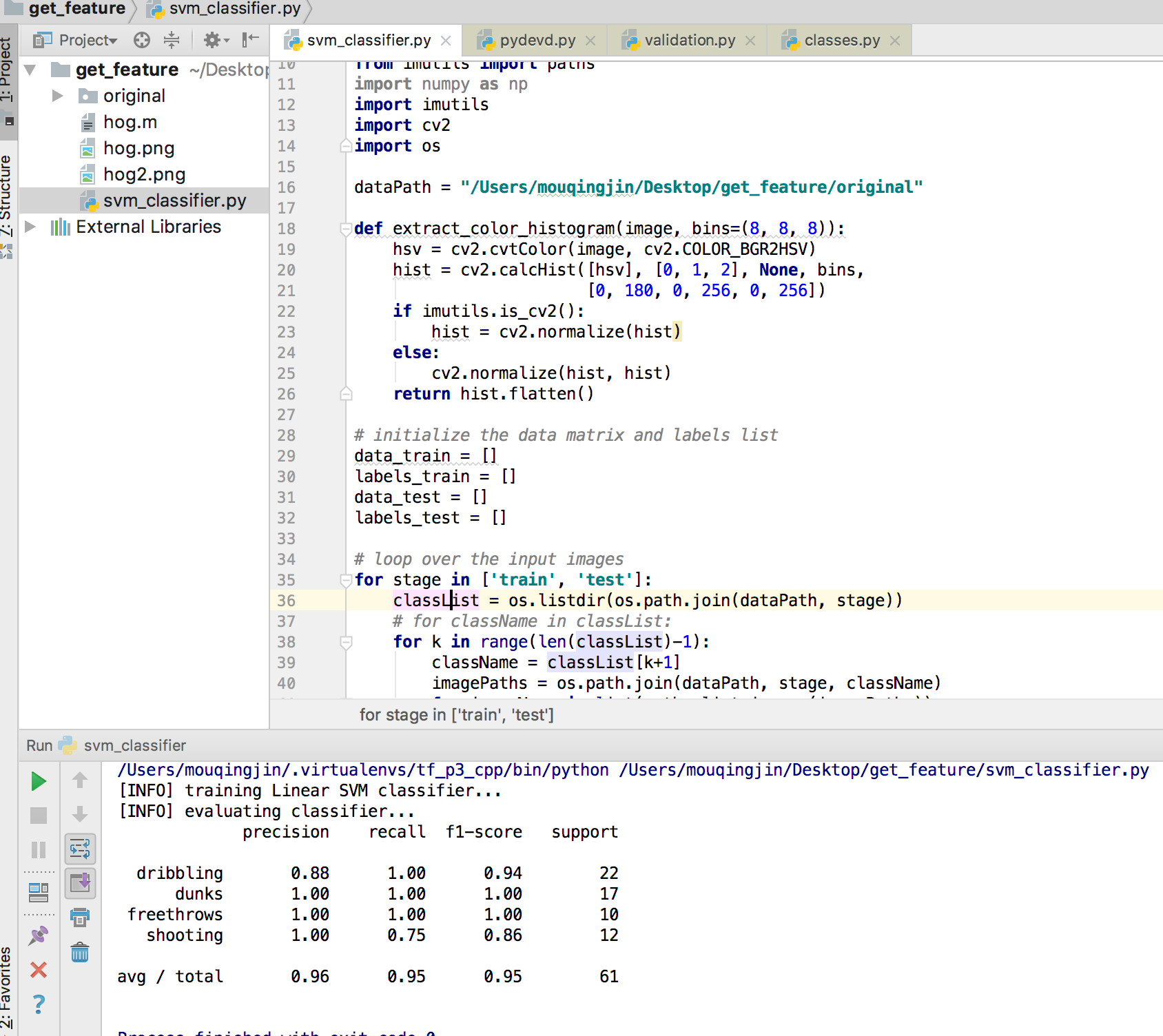
### **Hog (Histogram of oriented gradient) Feature**



### **LBP (Local Binary Patterns)**



### **Color histogram**

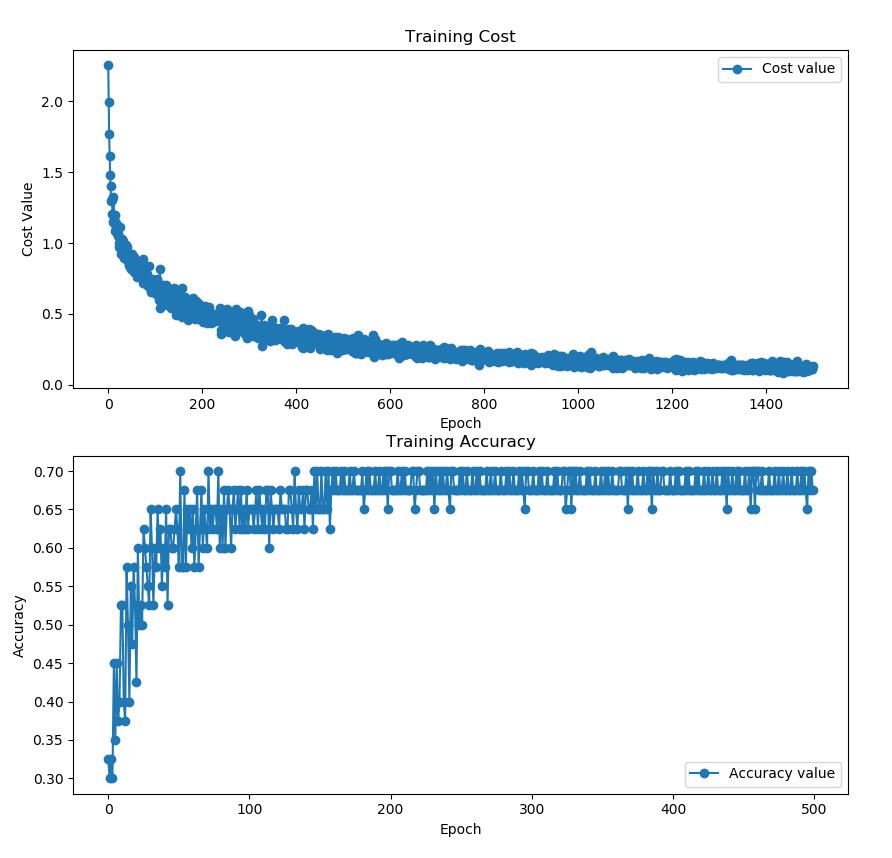


# **Conclusion**

Our deep leaning model gives more better results or accuracy on our provided dataset than our shallow learning models due to its learning ability.

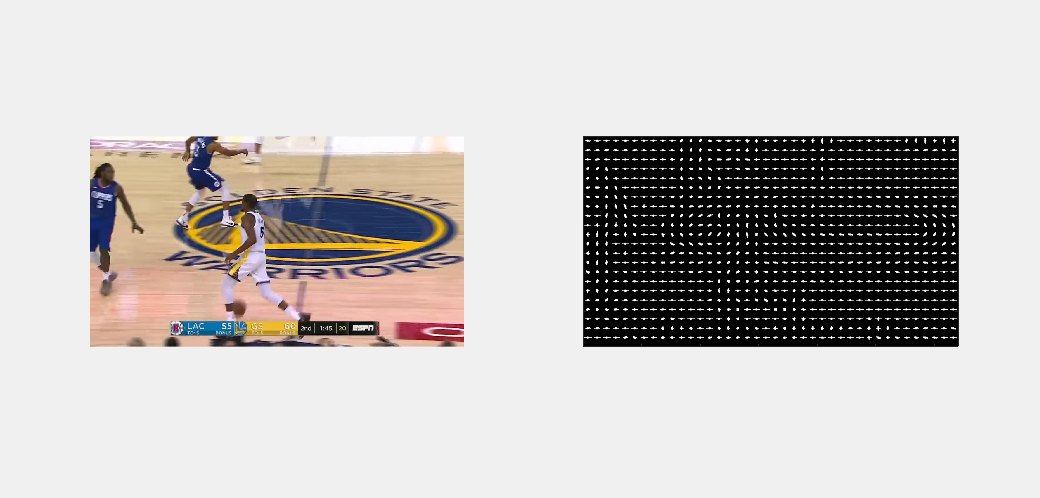
# **Appendix**

## **Regression:**



## **Training SVM with different feature**

### **Hog (Histogram of oriented gradient) Feature**



### **LBP (Local Binary Patterns**

