

## **Google Cloud Platform AutoML Vision**

機械学習 - イメージカラーの抽出









## 機械学習

### イメージカラー の抽出



### 01 Google Cloud Platform(GCP)

1-1 GCPの準備

1-2 プロジェクトの作成

#### 02 Auto ML Vision

2-1 Auto ML Visionの設定

2-2 データセット初期設定とバケット作成

2-3 学習用データの読み込み

### **03** 学習モデルの作成

3-1 学習モデルの作成

3-2 モデルのトレーニング

### 04 評価とテスト

4-1 評価

4-2 テスト

4-3 機械学習モデルで推論した結果





# Google Cloud Platform(GCP)





### 1-1 Google Cloud Platform(GCP)の準備

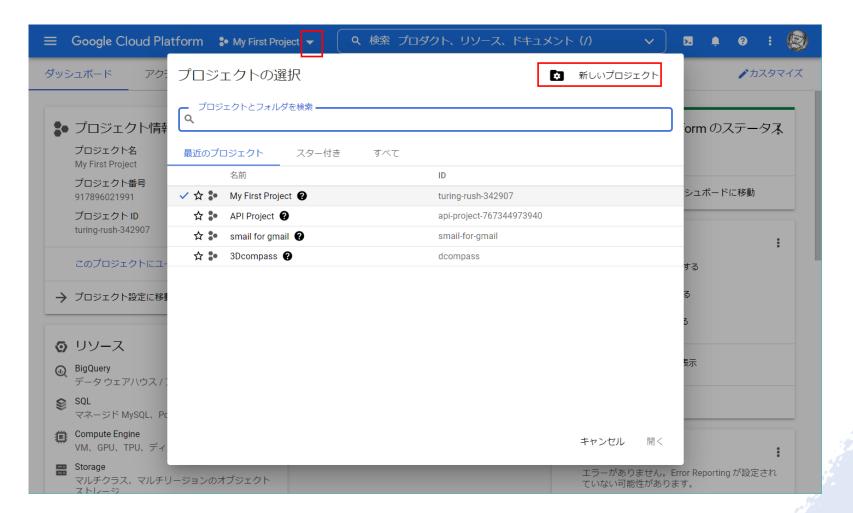
- ・AI画像分類ができるAutoML Visionを使うためには、Google Cloud Platform (GCP)にアカウントが必要です。
- <u>https://cloud.google.com/</u> から登録することでクレジット\$300が無料で付与されます。 GCPのコンソールを開いてアカウント、支払情報などを入力してアカウントを開設して ください。(詳細は、<u>Google Cloud Platform</u>を参照)
- ・初めてAutoML Visionを使用する場合もクレジット約\$170が無料で追加付与されます。 (2022/3現在)
- ※実験的に小規模な画像データ数、ラベルで動作確認を行う場合、無料分のクレジットで十分足りますが、課金予算オーバーにならないように注意して下さい。





#### 1-2 プロジェクトの作成

GCPにログインして、機械学習するためにプロジェクトを作成します。 「新しいプロジェクト」ー「プロジェクト名(任意名称)」を入力し「作成」を実行します。名前は任意で、プロジェクトIDは自動で付与されます。









空プロジェクトが作成されます。このプロジェクトにAI画像分類のライブラリやリソースを登録することになります。(プロジェクト単位で課金、ユーザ管理を行います。この段階で課金はされません)





### **Auto ML Vision**





### これから行う画像分類・分析の課題について

## 『ホームページのスクリーンショット 画像からイメージカラーを2色答えよ』

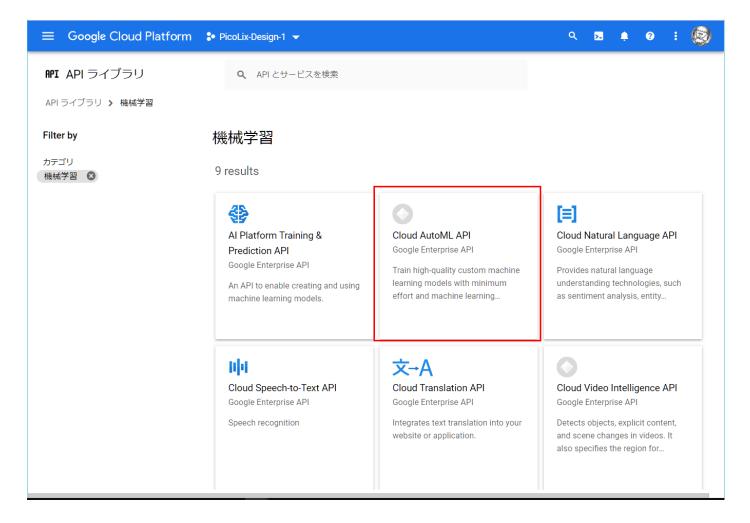
- ●画像を扱うライブラリーはAuto ML Visionです。 このライブラリーを使って、機械学習モデルを作成していきます。
- ●機械学習モデルの作成手順
  - •Auto ML Visionの設定
  - データセット初期設定とバケット作成
  - ・学習用データの読み込み
  - ・学習モデルの作成、評価、テスト



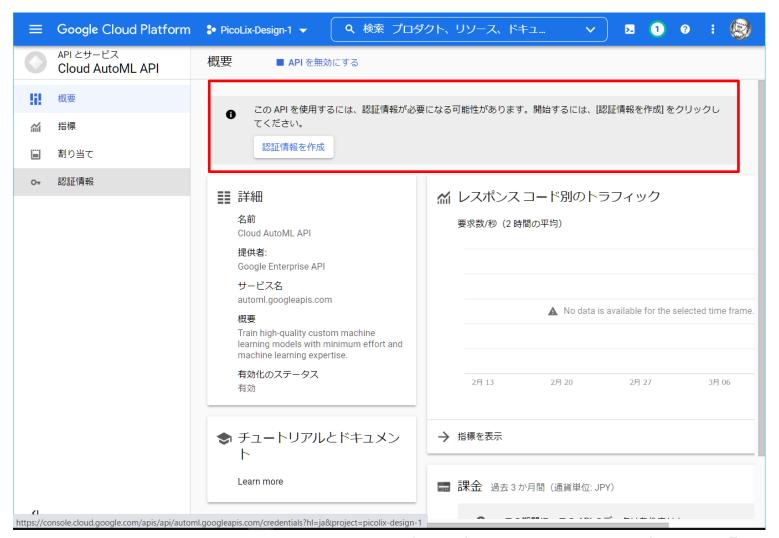
## 2.4

#### 2-1 Auto ML Visionの設定

作成したプロジェクトにAuto ML Visionライブラリーを割り当てます。 左上の「ナビゲーションメニュー」 - 「API-サービス」 - 「ライブラリー」 「機械学習」 - 「Cloud AutoML API」を選択して「有効にします」







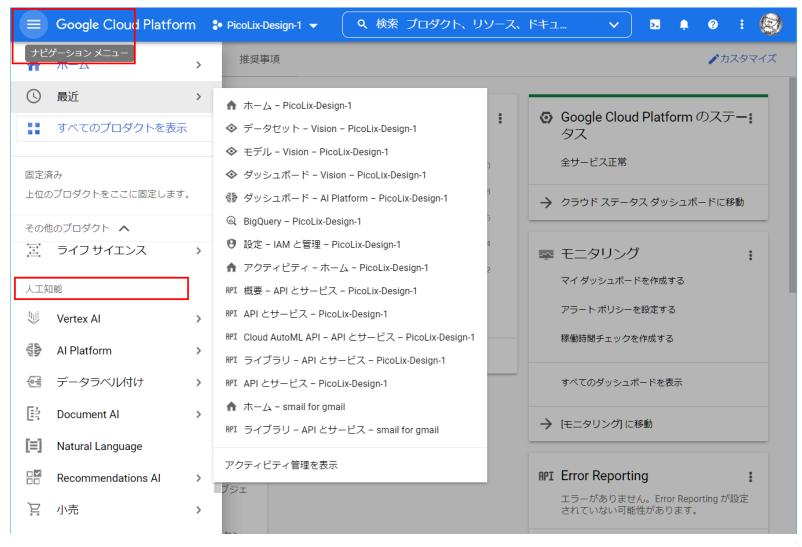
Cloud AutoML APIの概要画面で"認証情報が必要になる場合がある"と「認証情報を 作成」ボタンが表示されていますが、今回は認証情報を作成しません。

学習モデルのテスト確認はダッシュボード内でできますので確認だけなら必要ありま せん。(オンラインでAPI連携する場合は認証情報を作成する必要があります)



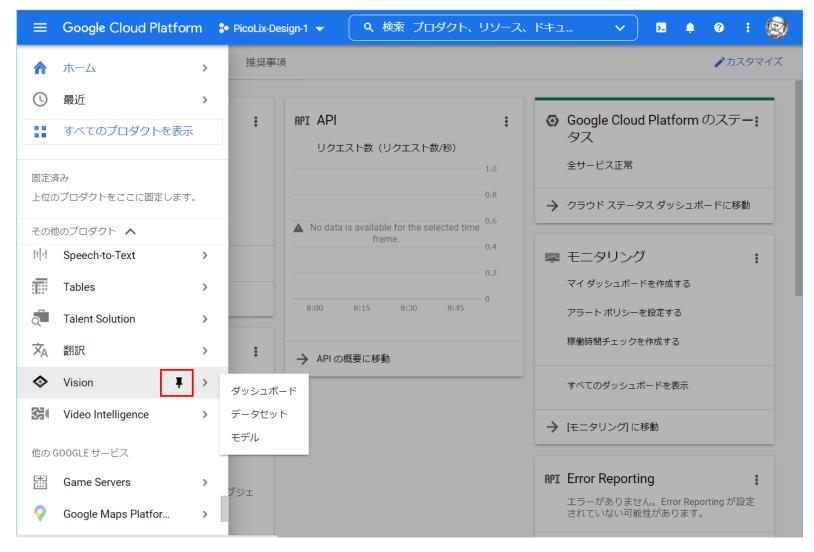


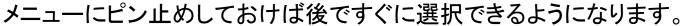
#### 左上の「ナビゲーションメニュー」ー「人工知能」ー「Vision」を選択。 さらにメニューを下にスクロールしてVisionを選択します。







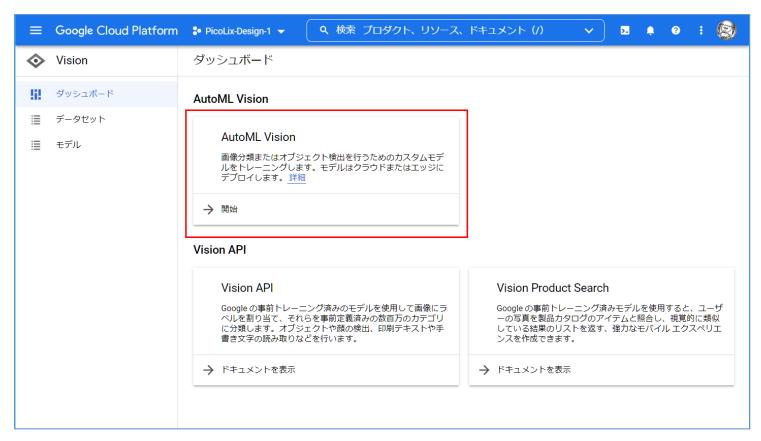








### 「ナビゲーションメニュー」 - 「Vision」 - 「ダッシュボード」を選択してAuto ML Visionの「開始」を実行します。



"Google Cloud Platform で課金を有効にしています..."と表示されます。プロジェクト「PicoLix-Design-1」の請求先アカウント設定します。(この段階ではまだ課金されません)





#### 2-2 データセット初期設定とバケット作成



データセットには画像(jpg)とそれに紐づくラベル名(csv)データが必要です。 まず、そのデータセットを保管するバケットと設定を行います。

・課題に対する必要なデータのホームページスクリーンショット画像(jpg)は私の運営サイトsearch.picolix.jpで保持している3万のホームページ画像と画像に紐づく2色のラベルデータ(csv)を利用しました。

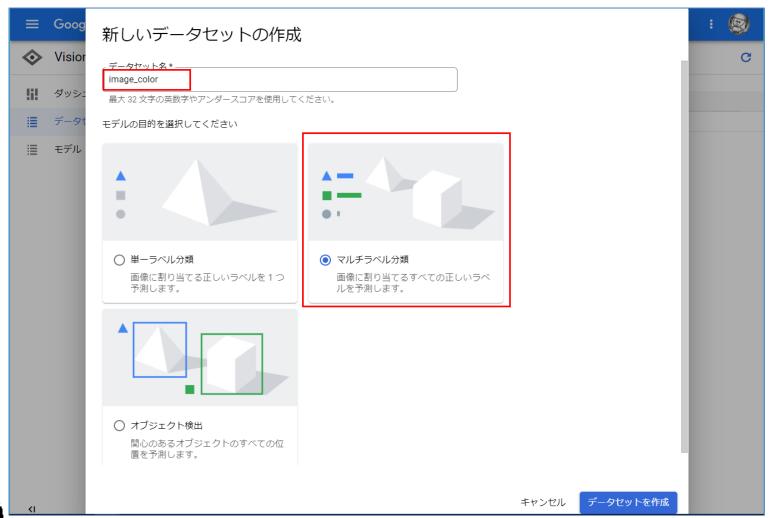
2色のラベルデータはイメージカラー計算値で自動取得しています。本来は人間の目で見た感じを設定すべきですが、総数が3万と多いことからこの手順を省きます。

判定率が落ちるかもしれませんがAuto ML Visionの画像分類モデルを作って機能するかを実証を優先します。

- ・ラベルデータ(csv)は、[Imageファイル名],[色1],[色2]の項目とします。[Imageファイル名],[色1],[色2],[色3]と3色の指定もできますが、学習トレーニングが長くなると予想されますので2色としています。
- ・イメージカラー計算参考プログラム

https://search.picolix.jp/dmoz/index.php?s\_keyword\_in=IT ←の例ではイメージカラーは計算により取得し登録されています。イメージカラー抽出プログラムはこちらです興味ある方はどうぞhttps://github.com/picolix/imagecolor

# 「ナビゲーションメニュー」ー「Vision」ー「データセット」 「新しいデータセット」を選択して任意のデータ名とモデルを選択します。ここでは、1画像につき1位の色名、2位の色名のラベルを用意するので、「マルチラベル分類」を選択します。



「ナビゲーションメニュー」ー「Cloud Strage」ー「ブラウザ」ー「バケットを作成」を実行します。 ※ここに学習したい画像ファイルとラベル用CSVファイルを保管することになります。 ディレクトリー構成は自由に作成できます。注意事項としてロケーションを正しく選択してお かないと、後でトレーニングするときに"ロケーションが違うために画像が移動できない"と エラーが発生します。

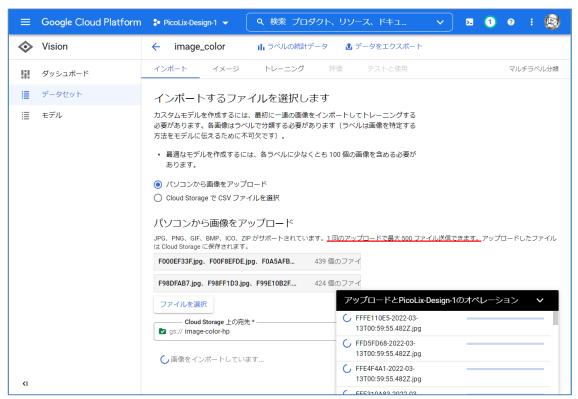




#### 2-3 学習用データの読み込み

「ナビゲーションメニュー」 - 「Vision」 - 「データセット」 - 「インポート」でバケットimage-color-hpに学習用画像(jpg)とラベル(csv)をアップロードします。

「インポート」画面から選択してアップ可能ですが、画像は1回500ファイルを複数セットアップした時に画像ファイル名にタイムスタンプの文字列が自動付与されてしまいCSV中で指定する画像ファイル名と異なってしまいます。従ってディレクトリー単位でまるごとアップして下さい。この場合はファイル名は変わりません。

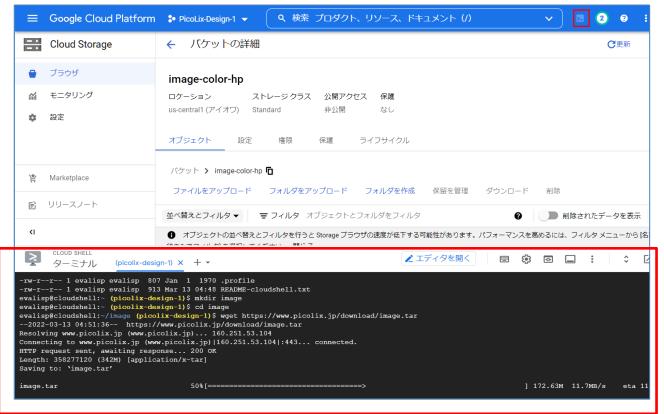


- ・1枚10kバイトの画像ファイルを800枚でだいたい5分で アップできます。
- ・1,000枚程度ならこの方法で良いのですが、万単位になると、時間がかかり過ぎてブラウザーが応答しなく無応答の警告がでます。アップロードは停止しているのか進んでいるのか分からなくなります。この場合cloud shellを使ってアップロードを行います。





#### Clouud shellを使って画像をアップロードする方法 右上のCloud shellアイコン[>\_]をクリックすると、ターミナルが起動します。



3万ファイルをtarでディレクト リーごとアーカイブしたファイ ルをCloud Strageに置くか、ど こかのWWWサーバに置きま す。

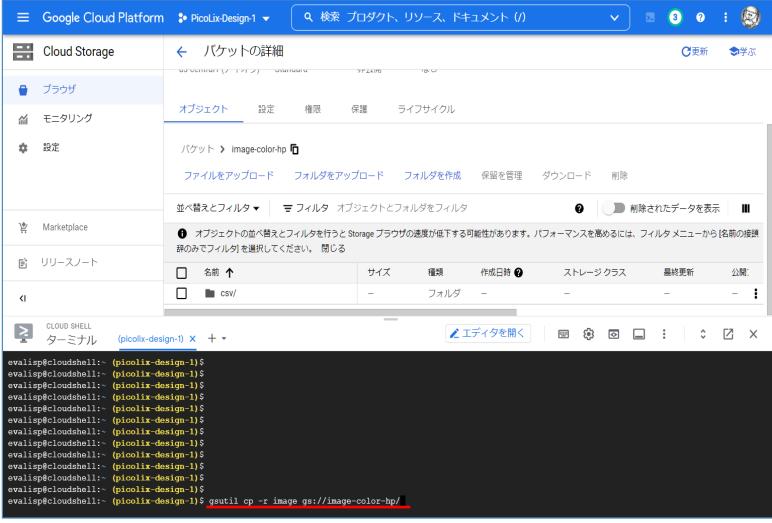
例ではWWWサーバに置い て、そこからwgetコマンドでダ ウンロードして、tarで解凍し ています。

※wget ,tar コマンドが利用できます



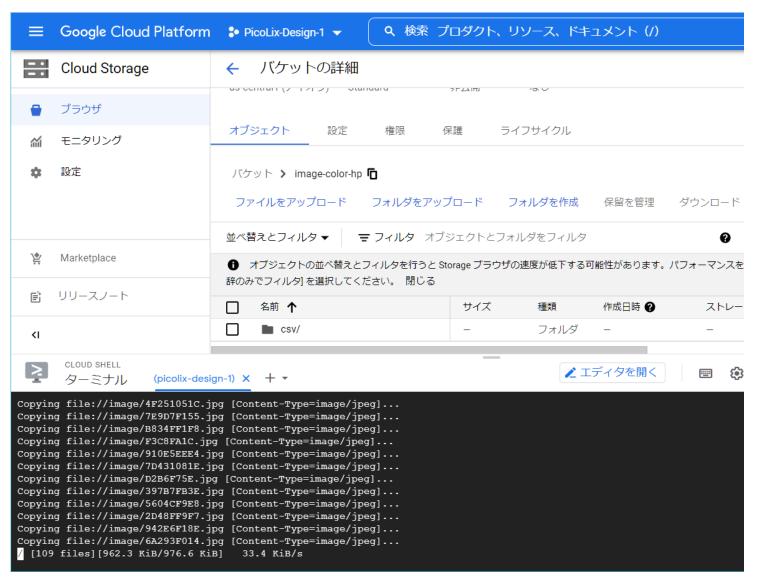


#### qutilコマンド cpでディレクトリーごとバケットにアップロードする。





qutil cp -r image gs://image-color-hp/ を実行してimage-color-hpのバケットに/imageフォルダーごと入れる。











#### •ラベル用CSVファイルについて

フォーマットは以下のようになります。

(詳細はこちらhttps://cloud.google.com/vision/automl/object-detection/docs/csv-format?hl=ja)

[TRAIN,TEST,VALIDATION],gs://バケット名,label1[,label2][,labelN]

TRAIN: この画像を使用してモデルをトレーニングします。

VALIDATION: 画像を使用して、トレーニング中にモデルが返す結果を検証します。

TEST: 画像を使用して、トレーニング後のモデルの結果を数値化します。

- ・LABEL名はラテン文字または数字で始まり、ラテン文字、数字、スペース、アンダースコア、ダッシュのみを含み、最大32文字まででなければない。UNICODEでも日本語名は使えない。
- ・バケットはAutoMLと同じ場所、同じストレージクラスの地域バケットを使用すること。
- ・各ラベルに少なくとも 10個のイメージが必要で、トレーニング用、検証用、テスト用の各セットに少なくとも 8個(TRAIN)、1個(VALIDATION)、1個(TEST) のイメージを割り当てる必要があります。

第1項目を外して、gs://image-color-hp/image/F5BCF8D1.jpg,orange,yellow\_greenのようにすると全データから適当に8:1:1に割り当ててくれます。

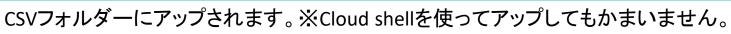




### •ラベル用CSVファイルのアップロード

作成したCSVファイルを選択して[続行]しアップロードします。

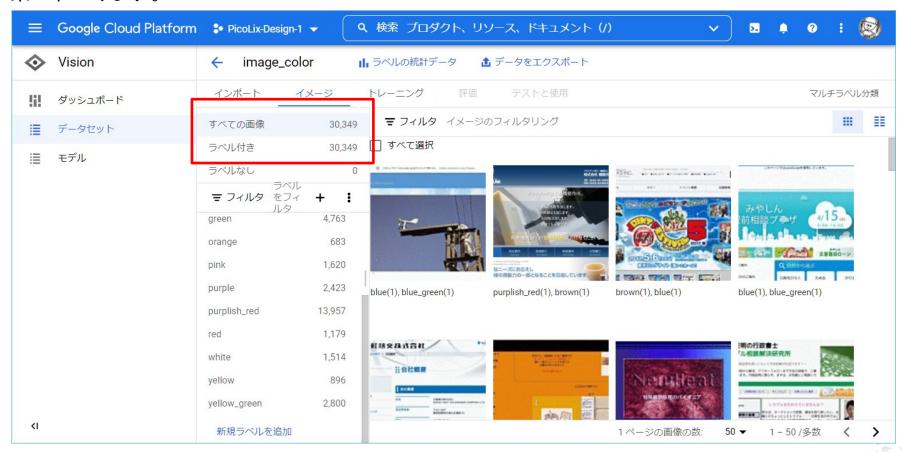
≡ Google Cloud Pl	atform 🛟 PicoLix-Design-1 ▼ 🔍 検索 プロダクト、リソース、ドキュメント (/) 🔻 🗘 🔞 🔞 🔞
Vision	← image_color 🕩 ラベルの統計データ 🏠 データをエクスポート
<b>ジ</b> ダッシュボード	インポート イメージ トレーニング 評価 テストと使用 マルチラベル分類 カスタムモデルを作成するには、最初に一連の画像をインポートしてトレーニングする
<b>≣</b> データセット	必要があります。各画像はラベルで分類する必要があります(ラベルは画像を特定する 方法をモデルに伝えるために不可欠です)。
<b>≣</b> モデル	<ul> <li>最適なモデルを作成するには、各ラベルに少なくとも100個の画像を含める必要があります。</li> <li>○ パソコンから画像をアップロード</li> <li>⑥ Cloud Storage で CSV ファイルを選択</li> <li>Cloud Storage で CSV ファイルを選択</li> <li>まだファイルをクラウドストレージ ピにアップロードしていない場合は、アップロードします。CSV ファイルは、画像への GCS パスのリストにする必要があります。指定できる画像形式は、JPG、PNG、GIF、BMP、ICO です。必要に応じて、トレーニング用、検証用、テスト用の分割を指定することもできます。</li> <li>CSV 形式のサンプル</li> </ul>
	<pre>[set,]image_path[,label1][,label2][,labelN] TRAIN,gs://My_Bucket/sample1.jpg.cat TEST,gs://My_Bucket/sample2.jpg,dog,happy</pre>
	gs:// *  image-color-hp/csv/train_gs_image-color-hp-en-2.csv 参照
<1	<b>続行</b>

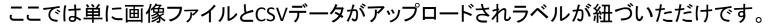




#### - イメージ画面

「ナビゲーションメニュー」ー「Vision」ー「データセット」ー「イメージ」を選択するとアップロード結果がわかります。

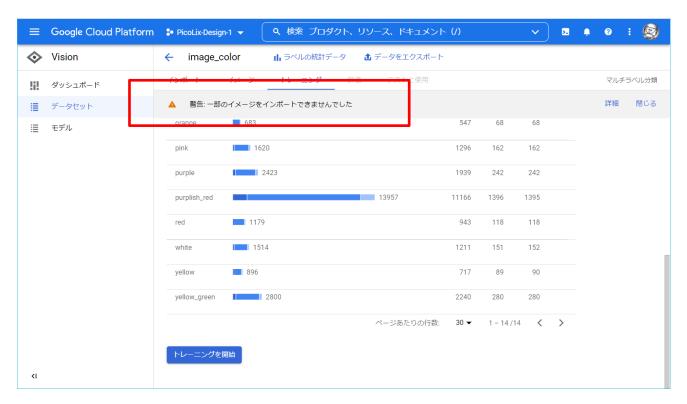




# 2'

#### •トレーニング画面

「ナビゲーションメニュー」ー「Vision」ー「データセット」ー「トレーニング」を選択するとアップロード結果がわかります。有効なデータ量と致命的なエラー及びエラーの数が多すぎなければ「トレーニングを開始」ボタンが有効になります。



画面で「一部のイメージをインポートできませんでした」と表示されている場合は、CSV中のファイル名が実際storage中になかった場合です。ファイル名が間違っているか画

ファイル名が間違っているか画像ファイルのアップロード漏れですのでその場合は再度アップしてください。

尚、数個欠落していてもエラー を無視して大丈夫です。

ここまででgoogle strageに 400Mバイト程使っています。 課金対象ですが、1円未満です。





## 学習モデルの作成

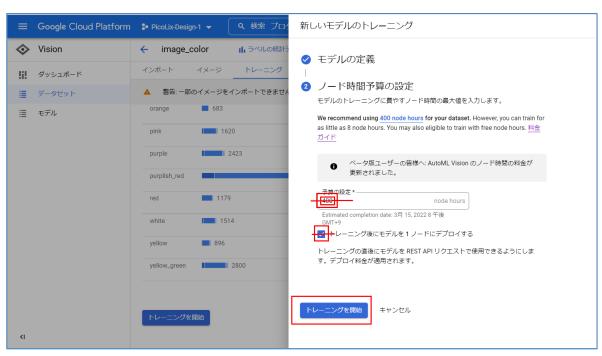




#### 3-1 学習モデルの作成

今回揃えたデータは、画像ファイル約3万ファイル、ラベルは1画像につき2つ、ラベル種類は16種のデータです。いよいよトレーニングですが、ここから本格的に課金が始まります。1ノード時間\$3.465 (2022/3時点)です。

#### 「トレーニングを開始」を実行します。



- 新しいモデルの定義をします。モデル・名称は適宜付けてください。
- ・ノード時間予算の設定 200を指定しました。もっと少ないデータ 量の場合は最低の8で十分です。 ※200を使い切った場合、\$600かかり ますので要注意です。
- ・トレーニング後にモデルを1ノード にデプロイはチェックを外す。(後で 手動でデプロイする) デプロイした場合はそのノード自体 が存在するだけで課金対象です。 放置していても課金されます。





#### 3-2 モデルのトレーニング

「新しいモデルをトレーニング」を実施します。トレーニングが終了するとこの画面からも分かりますが、終了メールが来ます。







画面の案内文には、"トレーニングには数時間かかることがあります。この時間には、ノードのトレーニング時間だけではなく、インフラストラクチャの設定と破棄も含まれますが、これらには料金がかかりません。"と表記されています。

この文章を読むとトレーニングには料金がかからないように思えますが、実際はトレーニングの時間ではなくてトレーニングの総ノード時間に課金されます。

例えば、1時間かかって10ノード時間を費やすのと、3時間かかって10ノード時間を費やした料金は両方とも10 ノード時間分(\$34.65)で同じということです。トレーニングが無償ということではありませんの注意してください。

予想では早くて6時間遅くても12時間ぐらいで終了するだろうと思っていましたが、18.5時間、140ノード時間かかってトレーニングが完了しました

≡	Google Cloud Platform	PicoLix-Design-1 ▼	Q 検索	プロダクト、リ	ソース、ドキュメント(	<i>n</i> ,	<b>&gt;</b>	. O :	
<b>\$</b>	Vision	データセット 🛨 新	fしいデータ	タセット					C
li!	ダッシュボード	● 名前	種類	イメージの総数	ラベル付きイメージの数	最終更新日時	ステータス	移行済み ②	
≣	データセット	C image_color	マル	30,349	30,349	2022/03/13 16:47:24	実行中: モ デルのト	いいえ	:
≣	モデル	ICN6451648908522160128	ルチラベル分類				グ		



# 31

#### トレーニングの課金状態

18.5時間、140ノード時間かかったトレーニングの課金状態は次のようになりました。フリーの無料分と AutoMLの無料分を食いつぶして、さらに別途6,796円の実費がかかりました。ここまでの一連の操作で55,764円です。140ノード時間分です。AutoML Image Classification Online Predictionsの無料分5,748円がありますが、これはオンライン予測にて使われるものです。

#### ●トレーニング前

<b>〒 フィルタ</b> クレジットをフィルタ					•	0 <u>+</u>
ウレジット名	ステータス 🕇	残りの割合	残りの値	元の値	種類	クレジット
Free Trial	❷ 利用可能	100%	¥34,483.03	¥34,484.00	1回限り	FreeTrial: 01B81A- C58445
▼ Cloud バンドル	✔ 利用可能	100%	¥20,232.00	¥20,232.00	1 回限り	
AutoML Image Classification Model Training	✔ 利用可能	100%	¥14,484.00	¥14,484.00	1回限り	AutoML
AutoML Image Classification Online Predictions	✔ 利用可能	100%	¥5,748.00	¥5,748.00	1回限り	AutoML

#### ●トレーニング前

<b>〒 フィルタ</b> クレジットをフィルタ							
クレジット名	ステータス 🛧	残りの割合	残りの値	元の値	種類	クレジット	
▼ Cloud バンドル	✔ 利用可能	28%	¥5,619.80	¥20,232.00	1回限り		
AutoML Image Classification Online Predictions	✔ 利用可能	98%	¥5,619.80	¥5,748.00	1回限り	AutoML_7f	
AutoML Image Classification Model Training	∅ 使用	0%	¥0.00	¥14,484.00	1回限り	AutoML_d8	
Free Trial	❷ 使用	0%	¥0.00	¥34,484.00	1回限り	FreeTrial:C	





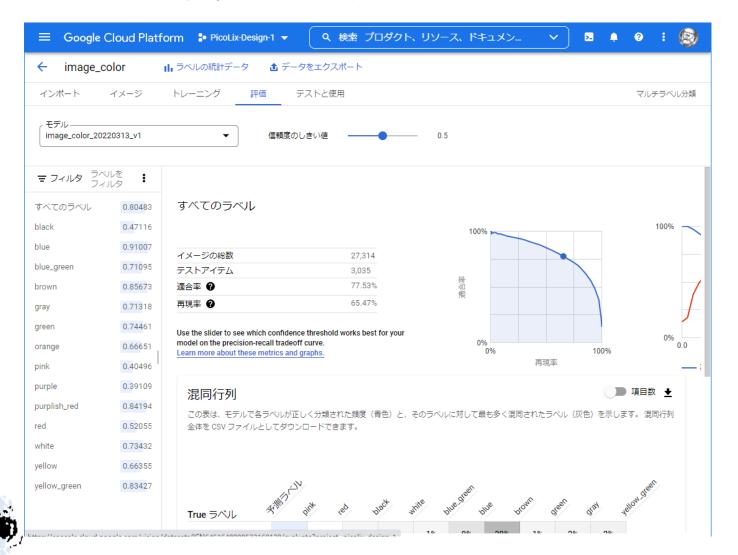
評価とテスト

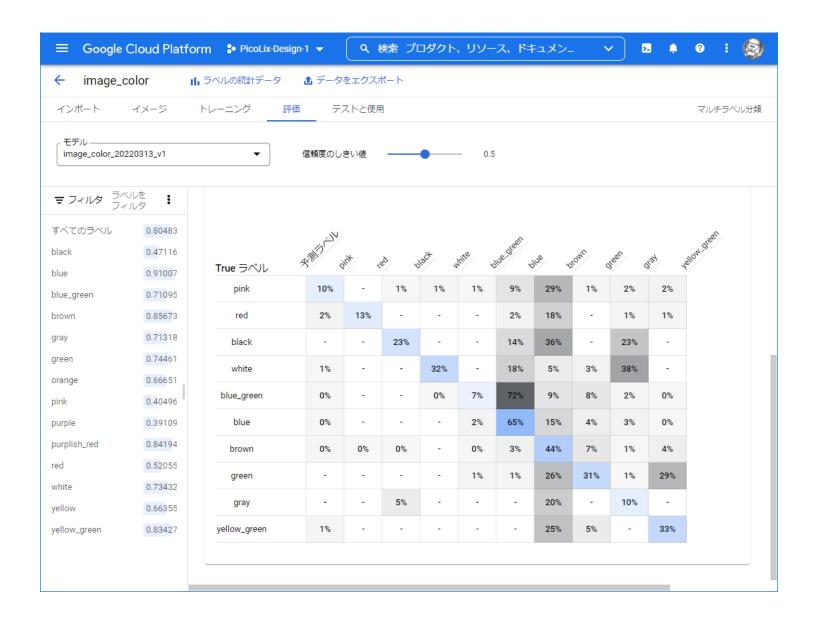


# 4

#### 4-1評価

#### トレーニングの結果は以下のようになりました。









平均適合率は、1.0に近いほどより良いデータモデルです。適合率と再現率は100%に近いほど良いです。 結果は**平均適合率0.805 適合率77.53% 再現率65.47%**となりました。

平均適合率は80.5%で90%を越えませんでしたがまずまずです。 適合率は抽出した2色がどれだけ判断が正解だったかの割合で77.53%で良いでしょう。 再現率は実際抽出した2色のデータの内どれだけ判断が正解だったかの割合で65.47%です。

もっと厳しい予想をしていたのですが評価できると思います。適合率と再現率はトレードオフの関係でどちらを優先するかはAutoMLでは設定できません。適合率と再現率の両方を向上させたい場合はより多くのデータをラベル付けすることになります。





## 4

#### 4-2 テスト

生成された学習モデルを使って、私のホームページのスクリーンショットでイメージカラーの抽出を試してみます。



https://www.picolix.jp

1.人間が判断した場合(私の場合) [赤]-[赤紫]

#### 2.イメージカラー抽出プログラム:

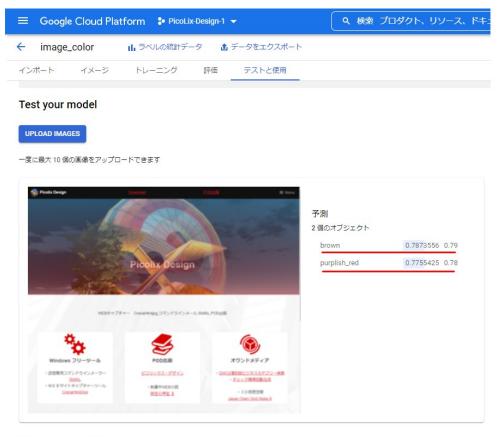
https://github.com/picolix/imagecolor での計算結果 [紫]-[赤紫]





# 4

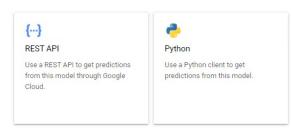
### 4-3機械学習モデルで推論した結果



イメージカラーはbrown[茶] - purplish\_red[赤紫]

brown(茶)が第一候補に出てきますが、これはもともとの16色データの精度が良くない為と思われます。それなりの答えが出ていますので良しとしましょう。このように、画像とラベルをAutoMLに喰わせることで、アルゴリズムやプログラムを全く気にすることなく、学習モデルを作ることができます。

#### Use your model



機械学習モデルをデプロイしている場合は、オンライン予測 リクエストが使えます。

システムを構築する時はこのRESET APIでリクエストし結果を 利用することになります。※テスト確認するだけであればこ の画面上で可能です。





画像データとラベルデータをAutoML Visionに投入するだけでノンプログラムミングで、ルールや特徴量などを全く気にすることなく実装できます。

皆さんも今後、機械学習を実装する場面がありましたらAutoML Visionを候補として検討してみてください。

