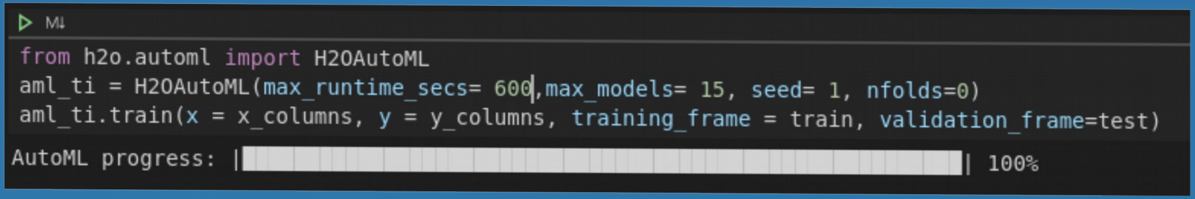
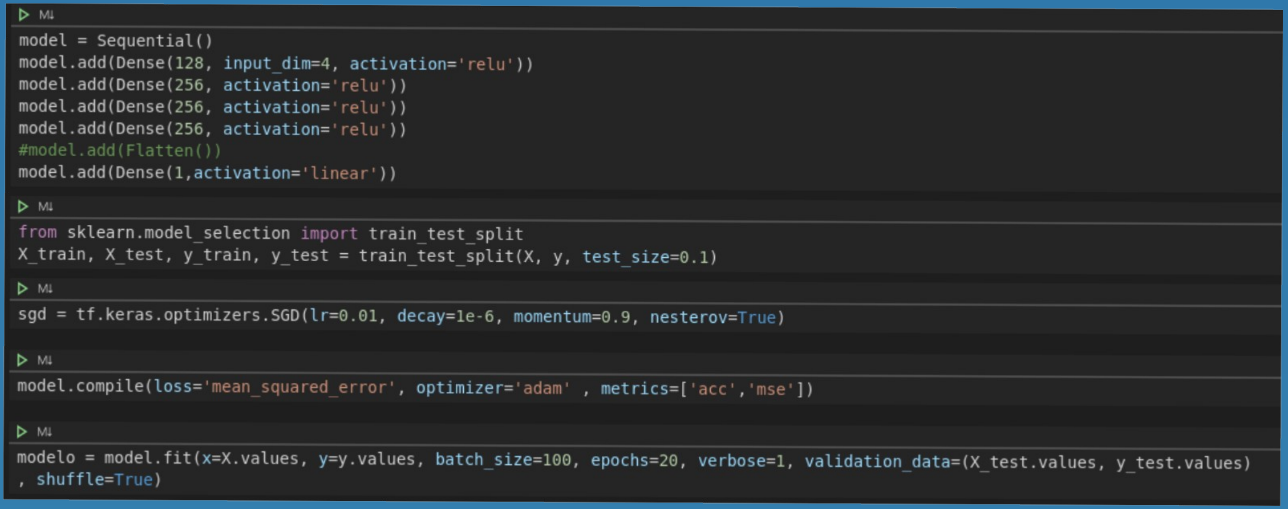


```
from h2o.automl import H2OAutoML
aml_ti = H2OAutoML(max_runtime_secs= 600,max_models= 15, seed= 1, nfolds=0)
aml_ti.train(x = x_columns, y = y_columns, training_frame = train, validation_frame=test)
```

```
AutoML progress: |████████████████████████████████████████████████████████| 100%
```

Diamond Kaggle Competition



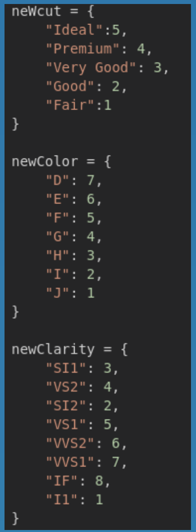
```
model = Sequential()
model.add(Dense(128, input_dim=4, activation='relu'))
model.add(Dense(256, activation='relu'))
model.add(Dense(256, activation='relu'))
model.add(Dense(256, activation='relu'))
#model.add(Flatten())
model.add(Dense(1,activation='linear'))
```

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.1)
```

```
sgd = tf.keras.optimizers.SGD(lr=0.01, decay=1e-6, momentum=0.9, nesterov=True)
```

```
model.compile(loss='mean_squared_error', optimizer='adam' , metrics=['acc','mse'])
```

```
modelo = model.fit(x=X.values, y=y.values, batch_size=100, epochs=20, verbose=1, validation_data=(X_test.values, y_test.values)
, shuffle=True)
```

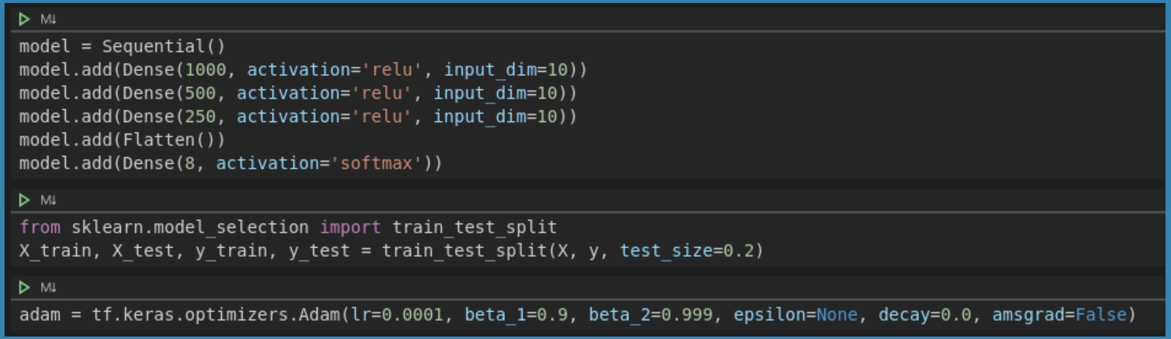


```
neWcut = {
    "Ideal":5,
    "Premium": 4,
    "Very Good": 3,
    "Good": 2,
    "Fair":1
}

newColor = {
    "D": 7,
    "E": 6,
    "F": 5,
    "G": 4,
    "H": 3,
    "I": 2,
    "J": 1
}

newClarity = {
    "SI1": 3,
    "VS2": 4,
    "SI2": 2,
    "VS1": 5,
    "VVS2": 6,
    "VVS1": 7,
    "IF": 8,
    "I1": 1
}
```

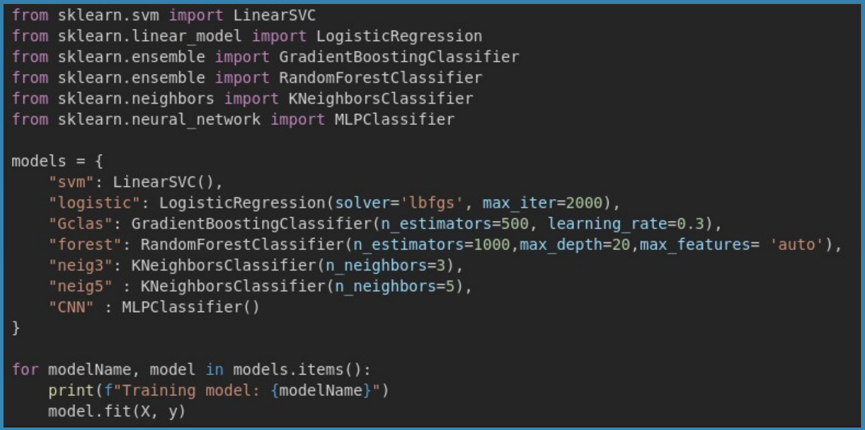
Avila Bible Kaggle Competition



```
model = Sequential()
model.add(Dense(1000, activation='relu', input_dim=10))
model.add(Dense(500, activation='relu', input_dim=10))
model.add(Dense(250, activation='relu', input_dim=10))
model.add(Flatten())
model.add(Dense(8, activation='softmax'))
```

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)
```

```
adam = tf.keras.optimizers.Adam(lr=0.0001, beta_1=0.9, beta_2=0.999, epsilon=None, decay=0.0, amsgrad=False)
```



```
from sklearn.svm import LinearSVC
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.neural_network import MLPClassifier

models = {
    "svm": LinearSVC(),
    "logistic": LogisticRegression(solver='lbfgs', max_iter=2000),
    "Gclas": GradientBoostingClassifier(n_estimators=500, learning_rate=0.3),
    "forest": RandomForestClassifier(n_estimators=1000,max_depth=20,max_features= 'auto'),
    "neig3": KNeighborsClassifier(n_neighbors=3),
    "neig5" : KNeighborsClassifier(n_neighbors=5),
    "CNN" : MLPClassifier()
}

for modelName, model in models.items():
    print(f"Training model: {modelName}")
    model.fit(X, y)
```

#KaggleCompetition