In [1]:

```
# Import our dependencies
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler,OneHotEncoder, MinMaxScaler
import pandas as pd
import tensorflow as tf
import numpy as np

# Import our input dataset
df = pd.read_csv('encoded_binned_df.csv')
df.head()
```

Out[1]:

	ERA	Hits	Earned Runs	Strike Outs	Home Runs	Wins	Losses	Outs Pitched	Batters Faced by Pitcher	Games Finished	Weight	Height	Games Started	salBin_low	salBin_
0	4.51	246	106	105	16	10	14	635	925	0	200	75	33	1	
1	5.97	37	23	25	0	0	5	104	162	0	185	75	7	1	
2	3.77	13	6	7	0	1	2	43	63	0	195	76	3	1	
3	4.53	214	95	82	20	7	18	566	797	0	178	71	31	1	
4	2.76	179	57	127	13	12	8	557	784	1	180	74	24	1	

In [2]:

```
### Drop unnecessary columns
df= df.filter(['Batters Faced by Pitcher','Outs Pitched','ERA','Strike Outs',"salBin_low","salBin_mid","s
df.head()
```

Out[2]:

	Batters Faced by Pitcher	Outs Pitched	ERA	Strike Outs	salBin_low	salBin_mid	salBin_high	salBin_top
0	925	635	4.51	105	1	0	0	0
1	162	104	5.97	25	1	0	0	0
2	63	43	3.77	7	1	0	0	0
3	797	566	4.53	82	1	0	0	0
4	784	557	2.76	127	1	0	0	0

Split Features/Target & Training/Testing Sets

Split into features and target

- **y variable**: Our target variables, `Salary-Bin_low`, `Salary-Bin_mid`, `Salary-Bin_high`, `Salary-Bin_top`
- **X variable**: Our features

In [3]:

```
# Split our preprocessed data into our features and target arrays
y = df[["salBin_low","salBin_mid","salBin_high","salBin_top"]].values
X = df.drop(["salBin_low","salBin_mid","salBin_high","salBin_top"],1).values

# Split the preprocessed data into a training and testing dataset
X_train, X_test, y_train, y_test = train_test_split(X, y, random_state=1)
```

```
C:\Users\alyss\anaconda3\envs\mlenv\lib\site-packages\ipykernel_launcher.py:3: FutureWarning: In a future
version of pandas all arguments of DataFrame.drop except for the argument 'labels' will be keyword-only
  This is separate from the ipykernel package so we can avoid doing imports until
```

Build and Instantiate `StandardScaler` object, then standardize numerical

features

In [4]:

```python
# Create a StandardScaler instance
scaler = StandardScaler()

# Fit the StandardScaler
X_scaler = scaler.fit(X_train)

# Scale the data
X_train_scaled = X_scaler.transform(X_train)
X_test_scaled = X_scaler.transform(X_test)
```

Build Neural Net Framework

In [34]:

```python
# Define the model - deep neural net
number_input_features = len(X_train[0])
hidden_nodes_layer1 = 128
hidden_nodes_layer2 = 64
hidden_nodes_layer3 = 24


nn = tf.keras.models.Sequential()

# First hidden layer
nn.add(
    tf.keras.layers.Dense(units=hidden_nodes_layer1, input_dim=number_input_features, activation="relu")
)

# Second hidden layer
nn.add(tf.keras.layers.Dense(units=hidden_nodes_layer2, activation="relu"))

# third hidden layer
nn.add(tf.keras.layers.Dense(units=hidden_nodes_layer3, activation="relu"))

# Output layer
nn.add(tf.keras.layers.Dense(units=4, activation="softmax"))

# Check the structure of the model
nn.summary()
```

```
Model: "sequential_8"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_26 (Dense)            (None, 128)               640

 dense_27 (Dense)            (None, 64)                8256

 dense_28 (Dense)            (None, 24)                1560

 dense_29 (Dense)            (None, 4)                 100

=================================================================
Total params: 10,556
Trainable params: 10,556
Non-trainable params: 0
_________________________________________________________________
```

Compile the Model

In [35]:

```python
# Compile the model
nn.compile(loss="kullback_leibler_divergence", optimizer="adam", metrics=["accuracy"])
```

Train the model

In [36]:

```
# Train the model
fit_model = nn.fit(X_train,y_train,epochs=100)
```

```
Epoch 1/100
116/116 [==============================] - 0s 696us/step - loss: 3.5779 - accuracy: 0.4076
Epoch 2/100
116/116 [==============================] - 0s 678us/step - loss: 2.9471 - accuracy: 0.4184
Epoch 3/100
116/116 [==============================] - 0s 687us/step - loss: 2.1938 - accuracy: 0.3922
Epoch 4/100
116/116 [==============================] - 0s 687us/step - loss: 1.4656 - accuracy: 0.4060
Epoch 5/100
116/116 [==============================] - 0s 704us/step - loss: 1.3717 - accuracy: 0.4165
Epoch 6/100
116/116 [==============================] - 0s 701us/step - loss: 1.4029 - accuracy: 0.4098
Epoch 7/100
116/116 [==============================] - 0s 670us/step - loss: 1.2957 - accuracy: 0.4314
Epoch 8/100
116/116 [==============================] - 0s 687us/step - loss: 1.4009 - accuracy: 0.4238
Epoch 9/100
116/116 [==============================] - 0s 687us/step - loss: 1.3292 - accuracy: 0.4408
Epoch 10/100
116/116 [==============================] - 0s 696us/step - loss: 1.4065 - accuracy: 0.4165
Epoch 11/100
116/116 [==============================] - 0s 704us/step - loss: 1.3307 - accuracy: 0.4287
Epoch 12/100
116/116 [==============================] - 0s 687us/step - loss: 1.2641 - accuracy: 0.4392
Epoch 13/100
116/116 [==============================] - 0s 696us/step - loss: 1.2885 - accuracy: 0.4344
Epoch 14/100
116/116 [==============================] - 0s 678us/step - loss: 1.3020 - accuracy: 0.4390
Epoch 15/100
116/116 [==============================] - 0s 687us/step - loss: 1.2541 - accuracy: 0.4422
Epoch 16/100
116/116 [==============================] - 0s 678us/step - loss: 1.2603 - accuracy: 0.4327
Epoch 17/100
116/116 [==============================] - 0s 687us/step - loss: 1.2831 - accuracy: 0.4327
Epoch 18/100
116/116 [==============================] - 0s 687us/step - loss: 1.2379 - accuracy: 0.4525
Epoch 19/100
116/116 [==============================] - 0s 704us/step - loss: 1.2518 - accuracy: 0.4325
Epoch 20/100
116/116 [==============================] - 0s 687us/step - loss: 1.2225 - accuracy: 0.4446
Epoch 21/100
116/116 [==============================] - 0s 687us/step - loss: 1.2598 - accuracy: 0.4368
Epoch 22/100
116/116 [==============================] - 0s 696us/step - loss: 1.2065 - accuracy: 0.4489
Epoch 23/100
116/116 [==============================] - 0s 696us/step - loss: 1.2111 - accuracy: 0.4546
Epoch 24/100
116/116 [==============================] - 0s 696us/step - loss: 1.2341 - accuracy: 0.4508
Epoch 25/100
116/116 [==============================] - 0s 696us/step - loss: 1.2306 - accuracy: 0.4479
Epoch 26/100
116/116 [==============================] - 0s 696us/step - loss: 1.2523 - accuracy: 0.4390
Epoch 27/100
116/116 [==============================] - 0s 704us/step - loss: 1.2061 - accuracy: 0.4381
Epoch 28/100
116/116 [==============================] - 0s 696us/step - loss: 1.2245 - accuracy: 0.4468
Epoch 29/100
116/116 [==============================] - 0s 704us/step - loss: 1.2252 - accuracy: 0.4487
Epoch 30/100
116/116 [==============================] - 0s 739us/step - loss: 1.2584 - accuracy: 0.4392
Epoch 31/100
116/116 [==============================] - 0s 696us/step - loss: 1.1914 - accuracy: 0.4503
Epoch 32/100
116/116 [==============================] - 0s 774us/step - loss: 1.1896 - accuracy: 0.4460
Epoch 33/100
116/116 [==============================] - 0s 730us/step - loss: 1.1828 - accuracy: 0.4522
Epoch 34/100
116/116 [==============================] - 0s 704us/step - loss: 1.1995 - accuracy: 0.4489
Epoch 35/100
```

```
116/116 [==============================] - 0s 696us/step - loss: 1.1872 - accuracy: 0.4543
Epoch 36/100
116/116 [==============================] - 0s 704us/step - loss: 1.2026 - accuracy: 0.4487
Epoch 37/100
116/116 [==============================] - 0s 704us/step - loss: 1.2075 - accuracy: 0.4481
Epoch 38/100
116/116 [==============================] - 0s 696us/step - loss: 1.1926 - accuracy: 0.4573
Epoch 39/100
116/116 [==============================] - 0s 722us/step - loss: 1.1866 - accuracy: 0.4557
Epoch 40/100
116/116 [==============================] - 0s 748us/step - loss: 1.1775 - accuracy: 0.4611
Epoch 41/100
116/116 [==============================] - 0s 748us/step - loss: 1.1715 - accuracy: 0.4503
Epoch 42/100
116/116 [==============================] - 0s 704us/step - loss: 1.1787 - accuracy: 0.4600
Epoch 43/100
116/116 [==============================] - 0s 687us/step - loss: 1.1639 - accuracy: 0.4687
Epoch 44/100
116/116 [==============================] - 0s 687us/step - loss: 1.1695 - accuracy: 0.4568
Epoch 45/100
116/116 [==============================] - 0s 696us/step - loss: 1.1619 - accuracy: 0.4649
Epoch 46/100
116/116 [==============================] - 0s 696us/step - loss: 1.1663 - accuracy: 0.4668
Epoch 47/100
116/116 [==============================] - 0s 696us/step - loss: 1.1665 - accuracy: 0.4584
Epoch 48/100
116/116 [==============================] - 0s 696us/step - loss: 1.1702 - accuracy: 0.4565
Epoch 49/100
116/116 [==============================] - 0s 696us/step - loss: 1.1717 - accuracy: 0.4581
Epoch 50/100
116/116 [==============================] - 0s 739us/step - loss: 1.1682 - accuracy: 0.4703
Epoch 51/100
116/116 [==============================] - 0s 704us/step - loss: 1.1600 - accuracy: 0.4714
Epoch 52/100
116/116 [==============================] - 0s 704us/step - loss: 1.1673 - accuracy: 0.4581
Epoch 53/100
116/116 [==============================] - 0s 704us/step - loss: 1.1579 - accuracy: 0.4695
Epoch 54/100
116/116 [==============================] - 0s 866us/step - loss: 1.1560 - accuracy: 0.4738
Epoch 55/100
116/116 [==============================] - 0s 748us/step - loss: 1.1579 - accuracy: 0.4711
Epoch 56/100
116/116 [==============================] - 0s 730us/step - loss: 1.1544 - accuracy: 0.4687
Epoch 57/100
116/116 [==============================] - 0s 678us/step - loss: 1.1511 - accuracy: 0.4692
Epoch 58/100
116/116 [==============================] - 0s 678us/step - loss: 1.1525 - accuracy: 0.4700
Epoch 59/100
116/116 [==============================] - 0s 696us/step - loss: 1.1515 - accuracy: 0.4738
Epoch 60/100
116/116 [==============================] - 0s 678us/step - loss: 1.1658 - accuracy: 0.4708
Epoch 61/100
116/116 [==============================] - 0s 687us/step - loss: 1.1494 - accuracy: 0.4776
Epoch 62/100
116/116 [==============================] - 0s 678us/step - loss: 1.1579 - accuracy: 0.4614
Epoch 63/100
116/116 [==============================] - 0s 696us/step - loss: 1.1653 - accuracy: 0.4692
Epoch 64/100
116/116 [==============================] - 0s 696us/step - loss: 1.1485 - accuracy: 0.4719
Epoch 65/100
116/116 [==============================] - 0s 687us/step - loss: 1.1498 - accuracy: 0.4714
Epoch 66/100
116/116 [==============================] - 0s 696us/step - loss: 1.1473 - accuracy: 0.4773
Epoch 67/100
116/116 [==============================] - 0s 687us/step - loss: 1.1489 - accuracy: 0.4749
Epoch 68/100
116/116 [==============================] - 0s 687us/step - loss: 1.1469 - accuracy: 0.4743
Epoch 69/100
116/116 [==============================] - 0s 687us/step - loss: 1.1487 - accuracy: 0.4733
Epoch 70/100
116/116 [==============================] - 0s 696us/step - loss: 1.1531 - accuracy: 0.4751
Epoch 71/100
116/116 [==============================] - 0s 696us/step - loss: 1.1544 - accuracy: 0.4762
Epoch 72/100
116/116 [==============================] - 0s 704us/step - loss: 1.1531 - accuracy: 0.4692
Epoch 73/100
```

```
116/116 [==============================] - 0s 704us/step - loss: 1.1477 - accuracy: 0.4803
Epoch 74/100
116/116 [==============================] - 0s 678us/step - loss: 1.1457 - accuracy: 0.4689
Epoch 75/100
116/116 [==============================] - 0s 678us/step - loss: 1.1454 - accuracy: 0.4770
Epoch 76/100
116/116 [==============================] - 0s 696us/step - loss: 1.1432 - accuracy: 0.4792
Epoch 77/100
116/116 [==============================] - 0s 696us/step - loss: 1.1420 - accuracy: 0.4762
Epoch 78/100
116/116 [==============================] - 0s 730us/step - loss: 1.1470 - accuracy: 0.4746
Epoch 79/100
116/116 [==============================] - 0s 713us/step - loss: 1.1482 - accuracy: 0.4797
Epoch 80/100
116/116 [==============================] - 0s 739us/step - loss: 1.1467 - accuracy: 0.4806
Epoch 81/100
116/116 [==============================] - 0s 687us/step - loss: 1.1433 - accuracy: 0.4776
Epoch 82/100
116/116 [==============================] - 0s 696us/step - loss: 1.1411 - accuracy: 0.4776
Epoch 83/100
116/116 [==============================] - 0s 713us/step - loss: 1.1525 - accuracy: 0.4749
Epoch 84/100
116/116 [==============================] - 0s 704us/step - loss: 1.1516 - accuracy: 0.4751
Epoch 85/100
116/116 [==============================] - 0s 704us/step - loss: 1.1486 - accuracy: 0.4692
Epoch 86/100
116/116 [==============================] - 0s 713us/step - loss: 1.1542 - accuracy: 0.4692
Epoch 87/100
116/116 [==============================] - 0s 687us/step - loss: 1.1485 - accuracy: 0.4738
Epoch 88/100
116/116 [==============================] - 0s 696us/step - loss: 1.1503 - accuracy: 0.4703
Epoch 89/100
116/116 [==============================] - 0s 713us/step - loss: 1.1432 - accuracy: 0.4803
Epoch 90/100
116/116 [==============================] - 0s 696us/step - loss: 1.1433 - accuracy: 0.4800
Epoch 91/100
116/116 [==============================] - 0s 704us/step - loss: 1.1444 - accuracy: 0.4816
Epoch 92/100
116/116 [==============================] - 0s 704us/step - loss: 1.1390 - accuracy: 0.4800
Epoch 93/100
116/116 [==============================] - 0s 696us/step - loss: 1.1414 - accuracy: 0.4827
Epoch 94/100
116/116 [==============================] - 0s 704us/step - loss: 1.1438 - accuracy: 0.4808
Epoch 95/100
116/116 [==============================] - 0s 696us/step - loss: 1.1450 - accuracy: 0.4800
Epoch 96/100
116/116 [==============================] - 0s 704us/step - loss: 1.1444 - accuracy: 0.4784
Epoch 97/100
116/116 [==============================] - 0s 696us/step - loss: 1.1436 - accuracy: 0.4776
Epoch 98/100
116/116 [==============================] - 0s 696us/step - loss: 1.1451 - accuracy: 0.4773
Epoch 99/100
116/116 [==============================] - 0s 708us/step - loss: 1.1434 - accuracy: 0.4778
Epoch 100/100
116/116 [==============================] - 0s 696us/step - loss: 1.1455 - accuracy: 0.4679
```

In [37]:

```python
# Evaluate the model using the test data
model_loss, model_accuracy = nn.evaluate(X_test_scaled,y_test,verbose=2)
print(f"Loss: {model_loss}, Accuracy: {model_accuracy*100:.2f}%")
```

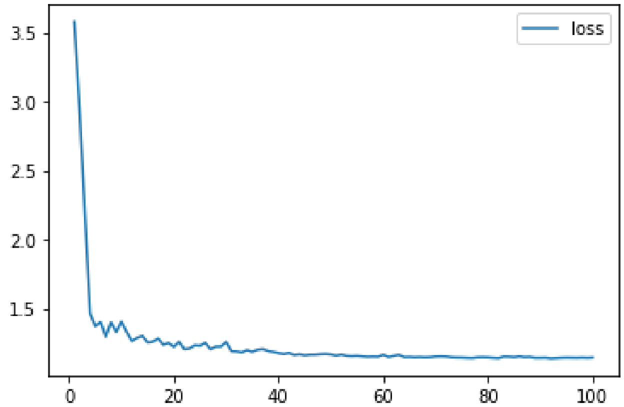
```
39/39 - 0s - loss: 1.3477 - accuracy: 0.3887 - 79ms/epoch - 2ms/step
Loss: 1.347665548324585, Accuracy: 38.87%
```

In [38]:

```python
# Create a DataFrame containing training history
history_df = pd.DataFrame(fit_model.history, index=range(1,len(fit_model.history["loss"])+1))

# Plot the loss
history_df.plot(y="loss")
```

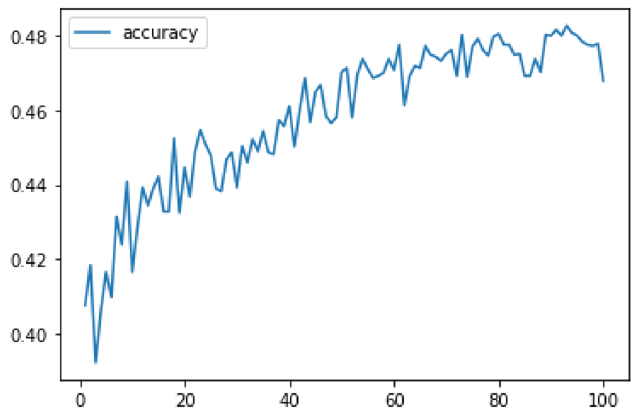
Out[38]: <AxesSubplot:>



In [39]:

```
# Plot the accuracy
history_df.plot(y="accuracy")
```

Out[39]: <AxesSubplot:>



In [ ]: