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('./pitcher_salaries_cleaned.csv')
df.head()
```

Out[1]:

	Year	Full Name	Age	Salary	ERA	Hits	Earned Runs	Strike Outs	Home Runs	Wins	Losses	Outs Pitched	Batters Faced by Pitcher	Games Finished	Weight	I
0	1990	AbbottJim	23	185000	4.51	246	106	105	16	10	14	635	925	0	200	
1	1990	AbbottPaul	23	100000	5.97	37	23	25	0	0	5	104	162	0	185	
2	1990	AldredScott	22	100000	3.77	13	6	7	0	1	2	43	63	0	195	
3	1990	AndersonAllan	26	300000	4.53	214	95	82	20	7	18	566	797	0	178	
4	1990	AppierKevin	23	100000	2.76	179	57	127	13	12	8	557	784	1	180	

Create Salary Brackets

In [2]:
```
# Look at distribution of salaries (suppressing scientific notation)
df['Salary'].describe().apply(lambda x: format(x, 'f'))
```

Out[2]:
```
count        4937.000000
mean      3011304.443387
std       4265619.190449
min        100000.000000
25%        327000.000000
50%        980000.000000
75%       4000000.000000
max      33000000.000000
Name: Salary, dtype: object
```

In [3]:
```
# create salary brackets and labels
bins = [0, 499999, 4999999, 9999999, 34999999]
labels = ['low', 'mid', 'high', 'top']
```

In [4]:
```
# apply salary brackets
df['salBin'] = pd.cut(df['Salary'], bins=bins, labels=labels)
df
```

Out[4]:

	Year	Full Name	Age	Salary	ERA	Hits	Earned Runs	Strike Outs	Home Runs	Wins	Losses	Outs Pitched	Batters Faced by Pitcher	Games Finished
0	1990	AbbottJim	23	185000	4.51	246	106	105	16	10	14	635	925	0
1	1990	AbbottPaul	23	100000	5.97	37	23	25	0	0	5	104	162	0
2	1990	AldredScott	22	100000	3.77	13	6	7	0	1	2	43	63	0
3	1990	AndersonAllan	26	300000	4.53	214	95	82	20	7	18	566	797	0
4	1990	AppierKevin	23	100000	2.76	179	57	127	13	12	8	557	784	1

	Year	Full Name	Age	Salary	ERA	Hits	Earned Runs	Strike Outs	Home Runs	Wins	Losses	Outs Pitched	Batters Faced by Pitcher	Games Finished
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
4932	2016	WorleyVance	29	2600000	3.53	84	34	56	11	2	2	260	365	13
4933	2016	WrightMike	26	510500	5.79	81	48	50	12	3	4	224	328	5
4934	2016	WrightSteven	32	514500	3.33	138	58	127	12	13	6	470	656	0
4935	2016	YoungChris	37	4250000	6.19	104	61	94	28	3	9	266	406	7
4936	2016	ZimmermannJordan	30	18000000	4.87	118	57	66	14	9	7	316	450	1

4937 rows × 20 columns

In [5]:

```
### Drop unnecessary columns
df= df.drop(["Full Name","Team","League","Age","Year","Salary"],1)
df.head()
```

```
C:\Users\alyss\anaconda3\envs\mlenv\lib\site-packages\ipykernel_launcher.py:2: FutureWarning: In a future
version of pandas all arguments of DataFrame.drop except for the argument 'labels' will be keyword-only
```

Out[5]:

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

Reduce number of rows

kept getting error in one-hot encoding, `ValueError: Buffer has wrong number of dimensions (expected 1, got 2)`

some suggested reducing sample size would solve issue (https://github.com/lmcinnes/umap/issues/496)

-- Update: reducing sample size did not solve issue with one-hot encoding

Encode Salary Bins column

In [6]:

```
# use get_dummies to one-hot encode the salarybin column
encoded_df=pd.get_dummies(df,columns=['salBin'],prefix="salBin")
encoded_df
```

Out[6]:

	ERA	Hits	Earned Runs	Strike Outs	Home Runs	Wins	Losses	Outs Pitched	Batters Faced by Pitcher	Games Finished	Weight	Height	Games Started	salBin_low	salE
0	4.51	246	106	105	16	10	14	635	925	0	200	75	33	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	salB
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	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
4932	3.53	84	34	56	11	2	2	260	365	13	240	74	4	0	
4933	5.79	81	48	50	12	3	4	224	328	5	240	78	12	0	
4934	3.33	138	58	127	12	13	6	470	656	0	215	74	24	0	
4935	6.19	104	61	94	28	3	9	266	406	7	255	82	13	0	
4936	4.87	118	57	66	14	9	7	316	450	1	225	74	18	0	

4937 rows × 17 columns

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 [7]:

```
# Split our preprocessed data into our features and target arrays
y = encoded_df[["salBin_low","salBin_mid","salBin_high","salBin_top"]].values
X = encoded_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 [8]:

```
# 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 [15]:

```
# Define the model - deep neural net
```

```python
number_input_features = len(X_train[0])
hidden_nodes_layer1 = 144
hidden_nodes_layer2 = 64
hidden_nodes_layer3 = 16


nn = tf.keras.models.Sequential()

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

# 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_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_4 (Dense)             (None, 144)               2016

 dense_5 (Dense)             (None, 64)                9280

 dense_6 (Dense)             (None, 16)                1040

 dense_7 (Dense)             (None, 4)                 68

=================================================================
Total params: 12,404
Trainable params: 12,404
Non-trainable params: 0
_________________________________________________________________
```

Compile the Model

In [16]:

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

Train the model

In [17]:

```python
# Train the model
fit_model = nn.fit(X_train,y_train,epochs=200)
```

```
Epoch 1/200
116/116 [==============================] - 0s 687us/step - loss: 1.2632 - accuracy: 0.3903
Epoch 2/200
116/116 [==============================] - 0s 704us/step - loss: 1.2010 - accuracy: 0.4130
Epoch 3/200
116/116 [==============================] - 0s 696us/step - loss: 1.2013 - accuracy: 0.4103
Epoch 4/200
116/116 [==============================] - 0s 687us/step - loss: 1.2011 - accuracy: 0.4130
Epoch 5/200
116/116 [==============================] - 0s 748us/step - loss: 1.2007 - accuracy: 0.4049
Epoch 6/200
116/116 [==============================] - 0s 748us/step - loss: 1.2008 - accuracy: 0.4084
Epoch 7/200
116/116 [==============================] - 0s 678us/step - loss: 1.2005 - accuracy: 0.4130
```

```
Epoch 8/200
116/116 [==============================] - 0s 704us/step - loss: 1.2010 - accuracy: 0.4044
Epoch 9/200
116/116 [==============================] - 0s 687us/step - loss: 1.2013 - accuracy: 0.4068
Epoch 10/200
116/116 [==============================] - 0s 704us/step - loss: 1.2006 - accuracy: 0.4033
Epoch 11/200
116/116 [==============================] - 0s 704us/step - loss: 1.2009 - accuracy: 0.4122
Epoch 12/200
116/116 [==============================] - 0s 696us/step - loss: 1.2011 - accuracy: 0.4130
Epoch 13/200
116/116 [==============================] - 0s 687us/step - loss: 1.2008 - accuracy: 0.4103
Epoch 14/200
116/116 [==============================] - 0s 696us/step - loss: 1.2009 - accuracy: 0.4130
Epoch 15/200
116/116 [==============================] - 0s 704us/step - loss: 1.2005 - accuracy: 0.4130
Epoch 16/200
116/116 [==============================] - 0s 713us/step - loss: 1.2014 - accuracy: 0.4130
Epoch 17/200
116/116 [==============================] - 0s 696us/step - loss: 1.2006 - accuracy: 0.4098
Epoch 18/200
116/116 [==============================] - 0s 896us/step - loss: 1.2020 - accuracy: 0.4087
Epoch 19/200
116/116 [==============================] - 0s 687us/step - loss: 1.2009 - accuracy: 0.4082
Epoch 20/200
116/116 [==============================] - 0s 704us/step - loss: 1.2014 - accuracy: 0.4130
Epoch 21/200
116/116 [==============================] - 0s 696us/step - loss: 1.2021 - accuracy: 0.3987
Epoch 22/200
116/116 [==============================] - 0s 687us/step - loss: 1.2006 - accuracy: 0.4090
Epoch 23/200
116/116 [==============================] - 0s 678us/step - loss: 1.2011 - accuracy: 0.4068
Epoch 24/200
116/116 [==============================] - 0s 678us/step - loss: 1.2011 - accuracy: 0.4130
Epoch 25/200
116/116 [==============================] - 0s 696us/step - loss: 1.2008 - accuracy: 0.4011
Epoch 26/200
116/116 [==============================] - 0s 696us/step - loss: 1.2009 - accuracy: 0.4127
Epoch 27/200
116/116 [==============================] - 0s 678us/step - loss: 1.2004 - accuracy: 0.4130
Epoch 28/200
116/116 [==============================] - 0s 670us/step - loss: 1.2003 - accuracy: 0.4098
Epoch 29/200
116/116 [==============================] - 0s 687us/step - loss: 1.2011 - accuracy: 0.4117
Epoch 30/200
116/116 [==============================] - 0s 687us/step - loss: 1.2009 - accuracy: 0.4114
Epoch 31/200
116/116 [==============================] - 0s 687us/step - loss: 1.2009 - accuracy: 0.4100
Epoch 32/200
116/116 [==============================] - 0s 713us/step - loss: 1.2007 - accuracy: 0.4127
Epoch 33/200
116/116 [==============================] - 0s 687us/step - loss: 1.2004 - accuracy: 0.4079
Epoch 34/200
116/116 [==============================] - 0s 696us/step - loss: 1.2000 - accuracy: 0.4149
Epoch 35/200
116/116 [==============================] - 0s 678us/step - loss: 1.2005 - accuracy: 0.4146
Epoch 36/200
116/116 [==============================] - 0s 687us/step - loss: 1.2005 - accuracy: 0.4133
Epoch 37/200
116/116 [==============================] - 0s 704us/step - loss: 1.2011 - accuracy: 0.4055
Epoch 38/200
116/116 [==============================] - 0s 687us/step - loss: 1.2012 - accuracy: 0.4057
Epoch 39/200
116/116 [==============================] - 0s 704us/step - loss: 1.2008 - accuracy: 0.4057
Epoch 40/200
116/116 [==============================] - 0s 696us/step - loss: 1.2006 - accuracy: 0.4130
Epoch 41/200
116/116 [==============================] - 0s 678us/step - loss: 1.2014 - accuracy: 0.4087
Epoch 42/200
116/116 [==============================] - 0s 687us/step - loss: 1.2008 - accuracy: 0.4079
Epoch 43/200
116/116 [==============================] - 0s 687us/step - loss: 1.2004 - accuracy: 0.4130
Epoch 44/200
116/116 [==============================] - 0s 687us/step - loss: 1.2009 - accuracy: 0.3971
Epoch 45/200
116/116 [==============================] - 0s 704us/step - loss: 1.2006 - accuracy: 0.4130
```

```
Epoch 46/200
116/116 [==============================] - 0s 696us/step - loss: 1.2004 - accuracy: 0.4136
Epoch 47/200
116/116 [==============================] - 0s 748us/step - loss: 1.2005 - accuracy: 0.4082
Epoch 48/200
116/116 [==============================] - 0s 678us/step - loss: 1.2013 - accuracy: 0.4082
Epoch 49/200
116/116 [==============================] - 0s 687us/step - loss: 1.2003 - accuracy: 0.4133
Epoch 50/200
116/116 [==============================] - 0s 687us/step - loss: 1.2007 - accuracy: 0.4130
Epoch 51/200
116/116 [==============================] - 0s 687us/step - loss: 1.2007 - accuracy: 0.4130
Epoch 52/200
116/116 [==============================] - 0s 696us/step - loss: 1.2007 - accuracy: 0.4065
Epoch 53/200
116/116 [==============================] - 0s 678us/step - loss: 1.1997 - accuracy: 0.4187
Epoch 54/200
116/116 [==============================] - 0s 678us/step - loss: 1.1828 - accuracy: 0.4673
Epoch 55/200
116/116 [==============================] - 0s 696us/step - loss: 1.1810 - accuracy: 0.4381
Epoch 56/200
116/116 [==============================] - 0s 704us/step - loss: 1.1681 - accuracy: 0.4776
Epoch 57/200
116/116 [==============================] - 0s 687us/step - loss: 1.1694 - accuracy: 0.4738
Epoch 58/200
116/116 [==============================] - 0s 687us/step - loss: 1.1673 - accuracy: 0.4795
Epoch 59/200
116/116 [==============================] - 0s 687us/step - loss: 1.1632 - accuracy: 0.4816
Epoch 60/200
116/116 [==============================] - 0s 678us/step - loss: 1.1733 - accuracy: 0.4719
Epoch 61/200
116/116 [==============================] - 0s 696us/step - loss: 1.1616 - accuracy: 0.4878
Epoch 62/200
116/116 [==============================] - 0s 696us/step - loss: 1.1629 - accuracy: 0.4860
Epoch 63/200
116/116 [==============================] - 0s 696us/step - loss: 1.1637 - accuracy: 0.4870
Epoch 64/200
116/116 [==============================] - 0s 678us/step - loss: 1.1634 - accuracy: 0.4833
Epoch 65/200
116/116 [==============================] - 0s 713us/step - loss: 1.1600 - accuracy: 0.4843
Epoch 66/200
116/116 [==============================] - 0s 687us/step - loss: 1.1629 - accuracy: 0.4811
Epoch 67/200
116/116 [==============================] - 0s 696us/step - loss: 1.1587 - accuracy: 0.4887
Epoch 68/200
116/116 [==============================] - 0s 696us/step - loss: 1.1611 - accuracy: 0.4849
Epoch 69/200
116/116 [==============================] - 0s 687us/step - loss: 1.1597 - accuracy: 0.4843
Epoch 70/200
116/116 [==============================] - 0s 678us/step - loss: 1.1725 - accuracy: 0.4541
Epoch 71/200
116/116 [==============================] - 0s 687us/step - loss: 1.1626 - accuracy: 0.4838
Epoch 72/200
116/116 [==============================] - 0s 704us/step - loss: 1.1612 - accuracy: 0.4857
Epoch 73/200
116/116 [==============================] - 0s 704us/step - loss: 1.1590 - accuracy: 0.4851
Epoch 74/200
116/116 [==============================] - 0s 687us/step - loss: 1.1633 - accuracy: 0.4841
Epoch 75/200
116/116 [==============================] - 0s 704us/step - loss: 1.1601 - accuracy: 0.4868
Epoch 76/200
116/116 [==============================] - 0s 704us/step - loss: 1.1579 - accuracy: 0.4857
Epoch 77/200
116/116 [==============================] - 0s 704us/step - loss: 1.1594 - accuracy: 0.4873
Epoch 78/200
116/116 [==============================] - 0s 694us/step - loss: 1.1596 - accuracy: 0.4884
Epoch 79/200
116/116 [==============================] - 0s 687us/step - loss: 1.1613 - accuracy: 0.4824
Epoch 80/200
116/116 [==============================] - 0s 696us/step - loss: 1.1601 - accuracy: 0.4862
Epoch 81/200
116/116 [==============================] - 0s 678us/step - loss: 1.1624 - accuracy: 0.4822
Epoch 82/200
116/116 [==============================] - 0s 696us/step - loss: 1.1587 - accuracy: 0.4827
Epoch 83/200
116/116 [==============================] - 0s 713us/step - loss: 1.1623 - accuracy: 0.4770
```

```
Epoch 84/200
116/116 [==============================] - 0s 713us/step - loss: 1.1622 - accuracy: 0.4816
Epoch 85/200
116/116 [==============================] - 0s 696us/step - loss: 1.1584 - accuracy: 0.4838
Epoch 86/200
116/116 [==============================] - 0s 678us/step - loss: 1.1616 - accuracy: 0.4822
Epoch 87/200
116/116 [==============================] - 0s 687us/step - loss: 1.1595 - accuracy: 0.4760
Epoch 88/200
116/116 [==============================] - 0s 687us/step - loss: 1.1589 - accuracy: 0.4862
Epoch 89/200
116/116 [==============================] - 0s 687us/step - loss: 1.1574 - accuracy: 0.4857
Epoch 90/200
116/116 [==============================] - 0s 704us/step - loss: 1.1659 - accuracy: 0.4719
Epoch 91/200
116/116 [==============================] - 0s 696us/step - loss: 1.1554 - accuracy: 0.4830
Epoch 92/200
116/116 [==============================] - 0s 696us/step - loss: 1.1583 - accuracy: 0.4822
Epoch 93/200
116/116 [==============================] - 0s 687us/step - loss: 1.1561 - accuracy: 0.4851
Epoch 94/200
116/116 [==============================] - 0s 678us/step - loss: 1.1604 - accuracy: 0.4716
Epoch 95/200
116/116 [==============================] - 0s 678us/step - loss: 1.1627 - accuracy: 0.4703
Epoch 96/200
116/116 [==============================] - 0s 704us/step - loss: 1.1539 - accuracy: 0.4884
Epoch 97/200
116/116 [==============================] - 0s 687us/step - loss: 1.1561 - accuracy: 0.4803
Epoch 98/200
116/116 [==============================] - 0s 678us/step - loss: 1.1530 - accuracy: 0.4884
Epoch 99/200
116/116 [==============================] - 0s 687us/step - loss: 1.1540 - accuracy: 0.4846
Epoch 100/200
116/116 [==============================] - 0s 687us/step - loss: 1.1579 - accuracy: 0.4692
Epoch 101/200
116/116 [==============================] - 0s 687us/step - loss: 1.1566 - accuracy: 0.4673
Epoch 102/200
116/116 [==============================] - 0s 678us/step - loss: 1.1531 - accuracy: 0.4827
Epoch 103/200
116/116 [==============================] - 0s 678us/step - loss: 1.1581 - accuracy: 0.4857
Epoch 104/200
116/116 [==============================] - 0s 670us/step - loss: 1.1544 - accuracy: 0.4857
Epoch 105/200
116/116 [==============================] - 0s 670us/step - loss: 1.1607 - accuracy: 0.4646
Epoch 106/200
116/116 [==============================] - 0s 687us/step - loss: 1.1508 - accuracy: 0.4787
Epoch 107/200
116/116 [==============================] - 0s 696us/step - loss: 1.1455 - accuracy: 0.4862
Epoch 108/200
116/116 [==============================] - 0s 922us/step - loss: 1.1452 - accuracy: 0.4843
Epoch 109/200
116/116 [==============================] - 0s 704us/step - loss: 1.1495 - accuracy: 0.4843
Epoch 110/200
116/116 [==============================] - 0s 708us/step - loss: 1.1497 - accuracy: 0.4897
Epoch 111/200
116/116 [==============================] - 0s 704us/step - loss: 1.1516 - accuracy: 0.4746
Epoch 112/200
116/116 [==============================] - 0s 687us/step - loss: 1.1480 - accuracy: 0.4873
Epoch 113/200
116/116 [==============================] - 0s 696us/step - loss: 1.1473 - accuracy: 0.4835
Epoch 114/200
116/116 [==============================] - 0s 713us/step - loss: 1.1470 - accuracy: 0.4841
Epoch 115/200
116/116 [==============================] - 0s 748us/step - loss: 1.1447 - accuracy: 0.4905
Epoch 116/200
116/116 [==============================] - 0s 696us/step - loss: 1.1575 - accuracy: 0.4646
Epoch 117/200
116/116 [==============================] - 0s 687us/step - loss: 1.1504 - accuracy: 0.4716
Epoch 118/200
116/116 [==============================] - 0s 678us/step - loss: 1.1482 - accuracy: 0.4657
Epoch 119/200
116/116 [==============================] - 0s 678us/step - loss: 1.1470 - accuracy: 0.4830
Epoch 120/200
116/116 [==============================] - 0s 713us/step - loss: 1.1449 - accuracy: 0.4838
Epoch 121/200
116/116 [==============================] - 0s 696us/step - loss: 1.1477 - accuracy: 0.4797
```

```
Epoch 122/200
116/116 [==============================] - 0s 704us/step - loss: 1.1455 - accuracy: 0.4846
Epoch 123/200
116/116 [==============================] - 0s 687us/step - loss: 1.1449 - accuracy: 0.4851
Epoch 124/200
116/116 [==============================] - 0s 696us/step - loss: 1.1468 - accuracy: 0.4851
Epoch 125/200
116/116 [==============================] - 0s 696us/step - loss: 1.1391 - accuracy: 0.4876
Epoch 126/200
116/116 [==============================] - 0s 696us/step - loss: 1.1420 - accuracy: 0.4838
Epoch 127/200
116/116 [==============================] - 0s 730us/step - loss: 1.1464 - accuracy: 0.4830
Epoch 128/200
116/116 [==============================] - 0s 687us/step - loss: 1.1460 - accuracy: 0.4841
Epoch 129/200
116/116 [==============================] - 0s 731us/step - loss: 1.1443 - accuracy: 0.4822
Epoch 130/200
116/116 [==============================] - 0s 704us/step - loss: 1.1431 - accuracy: 0.4824
Epoch 131/200
116/116 [==============================] - 0s 739us/step - loss: 1.1449 - accuracy: 0.4830
Epoch 132/200
116/116 [==============================] - 0s 748us/step - loss: 1.1446 - accuracy: 0.4824
Epoch 133/200
116/116 [==============================] - 0s 704us/step - loss: 1.1455 - accuracy: 0.4849
Epoch 134/200
116/116 [==============================] - 0s 687us/step - loss: 1.1452 - accuracy: 0.4851
Epoch 135/200
116/116 [==============================] - 0s 722us/step - loss: 1.1406 - accuracy: 0.4857
Epoch 136/200
116/116 [==============================] - 0s 948us/step - loss: 1.1458 - accuracy: 0.4806
Epoch 137/200
116/116 [==============================] - 0s 722us/step - loss: 1.1477 - accuracy: 0.4765
Epoch 138/200
116/116 [==============================] - 0s 704us/step - loss: 1.1433 - accuracy: 0.4846
Epoch 139/200
116/116 [==============================] - 0s 704us/step - loss: 1.1403 - accuracy: 0.4873
Epoch 140/200
116/116 [==============================] - 0s 704us/step - loss: 1.1392 - accuracy: 0.4862
Epoch 141/200
116/116 [==============================] - 0s 713us/step - loss: 1.1384 - accuracy: 0.4878
Epoch 142/200
116/116 [==============================] - 0s 696us/step - loss: 1.1395 - accuracy: 0.4824
Epoch 143/200
116/116 [==============================] - 0s 696us/step - loss: 1.1388 - accuracy: 0.4835
Epoch 144/200
116/116 [==============================] - 0s 713us/step - loss: 1.1394 - accuracy: 0.4851
Epoch 145/200
116/116 [==============================] - 0s 757us/step - loss: 1.1427 - accuracy: 0.4860
Epoch 146/200
116/116 [==============================] - 0s 687us/step - loss: 1.1401 - accuracy: 0.4843
Epoch 147/200
116/116 [==============================] - 0s 704us/step - loss: 1.1428 - accuracy: 0.4792
Epoch 148/200
116/116 [==============================] - 0s 704us/step - loss: 1.1401 - accuracy: 0.4916
Epoch 149/200
116/116 [==============================] - 0s 774us/step - loss: 1.1422 - accuracy: 0.4838
Epoch 150/200
116/116 [==============================] - 0s 757us/step - loss: 1.1388 - accuracy: 0.4895
Epoch 151/200
116/116 [==============================] - 0s 739us/step - loss: 1.1396 - accuracy: 0.4900
Epoch 152/200
116/116 [==============================] - 0s 687us/step - loss: 1.1435 - accuracy: 0.4843
Epoch 153/200
116/116 [==============================] - 0s 713us/step - loss: 1.1404 - accuracy: 0.4819
Epoch 154/200
116/116 [==============================] - 0s 774us/step - loss: 1.1430 - accuracy: 0.4895
Epoch 155/200
116/116 [==============================] - 0s 757us/step - loss: 1.1408 - accuracy: 0.4873
Epoch 156/200
116/116 [==============================] - 0s 713us/step - loss: 1.1430 - accuracy: 0.4703
Epoch 157/200
116/116 [==============================] - 0s 687us/step - loss: 1.1436 - accuracy: 0.4795
Epoch 158/200
116/116 [==============================] - 0s 765us/step - loss: 1.1475 - accuracy: 0.4843
Epoch 159/200
116/116 [==============================] - 0s 757us/step - loss: 1.1413 - accuracy: 0.4838
```

```
Epoch 160/200
116/116 [==============================] - 0s 713us/step - loss: 1.1358 - accuracy: 0.4838
Epoch 161/200
116/116 [==============================] - 0s 696us/step - loss: 1.1416 - accuracy: 0.4843
Epoch 162/200
116/116 [==============================] - 0s 713us/step - loss: 1.1409 - accuracy: 0.4873
Epoch 163/200
116/116 [==============================] - 0s 748us/step - loss: 1.1386 - accuracy: 0.4849
Epoch 164/200
116/116 [==============================] - 0s 730us/step - loss: 1.1385 - accuracy: 0.4870
Epoch 165/200
116/116 [==============================] - 0s 696us/step - loss: 1.1398 - accuracy: 0.4827
Epoch 166/200
116/116 [==============================] - 0s 713us/step - loss: 1.1409 - accuracy: 0.4800
Epoch 167/200
116/116 [==============================] - 0s 774us/step - loss: 1.1484 - accuracy: 0.4768
Epoch 168/200
116/116 [==============================] - 0s 748us/step - loss: 1.1423 - accuracy: 0.4784
Epoch 169/200
116/116 [==============================] - 0s 713us/step - loss: 1.1425 - accuracy: 0.4708
Epoch 170/200
116/116 [==============================] - 0s 696us/step - loss: 1.1443 - accuracy: 0.4849
Epoch 171/200
116/116 [==============================] - 0s 722us/step - loss: 1.1428 - accuracy: 0.4819
Epoch 172/200
116/116 [==============================] - 0s 774us/step - loss: 1.1376 - accuracy: 0.4824
Epoch 173/200
116/116 [==============================] - 0s 852us/step - loss: 1.1417 - accuracy: 0.4857
Epoch 174/200
116/116 [==============================] - 0s 713us/step - loss: 1.1407 - accuracy: 0.4827
Epoch 175/200
116/116 [==============================] - 0s 757us/step - loss: 1.1431 - accuracy: 0.4811
Epoch 176/200
116/116 [==============================] - 0s 748us/step - loss: 1.1370 - accuracy: 0.4792
Epoch 177/200
116/116 [==============================] - 0s 713us/step - loss: 1.1451 - accuracy: 0.4814
Epoch 178/200
116/116 [==============================] - 0s 678us/step - loss: 1.1391 - accuracy: 0.4830
Epoch 179/200
116/116 [==============================] - 0s 704us/step - loss: 1.1368 - accuracy: 0.4816
Epoch 180/200
116/116 [==============================] - 0s 713us/step - loss: 1.1355 - accuracy: 0.4838
Epoch 181/200
116/116 [==============================] - 0s 713us/step - loss: 1.1393 - accuracy: 0.4843
Epoch 182/200
116/116 [==============================] - 0s 696us/step - loss: 1.1397 - accuracy: 0.4878
Epoch 183/200
116/116 [==============================] - 0s 687us/step - loss: 1.1395 - accuracy: 0.4733
Epoch 184/200
116/116 [==============================] - 0s 713us/step - loss: 1.1388 - accuracy: 0.4824
Epoch 185/200
116/116 [==============================] - 0s 704us/step - loss: 1.1378 - accuracy: 0.4865
Epoch 186/200
116/116 [==============================] - 0s 696us/step - loss: 1.1426 - accuracy: 0.4824
Epoch 187/200
116/116 [==============================] - 0s 687us/step - loss: 1.1391 - accuracy: 0.4830
Epoch 188/200
116/116 [==============================] - 0s 687us/step - loss: 1.1416 - accuracy: 0.4849
Epoch 189/200
116/116 [==============================] - 0s 713us/step - loss: 1.1399 - accuracy: 0.4827
Epoch 190/200
116/116 [==============================] - 0s 696us/step - loss: 1.1446 - accuracy: 0.4814
Epoch 191/200
116/116 [==============================] - 0s 713us/step - loss: 1.1412 - accuracy: 0.4857
Epoch 192/200
116/116 [==============================] - 0s 687us/step - loss: 1.1390 - accuracy: 0.4860
Epoch 193/200
116/116 [==============================] - 0s 704us/step - loss: 1.1418 - accuracy: 0.4762
Epoch 194/200
116/116 [==============================] - 0s 687us/step - loss: 1.1454 - accuracy: 0.4803
Epoch 195/200
116/116 [==============================] - 0s 730us/step - loss: 1.1369 - accuracy: 0.4838
Epoch 196/200
116/116 [==============================] - 0s 774us/step - loss: 1.1345 - accuracy: 0.4835
Epoch 197/200
116/116 [==============================] - 0s 765us/step - loss: 1.1398 - accuracy: 0.4824
```

```
Epoch 198/200
116/116 [==============================] - 0s 704us/step - loss: 1.1396 - accuracy: 0.4806
Epoch 199/200
116/116 [==============================] - 0s 691us/step - loss: 1.1404 - accuracy: 0.4833
Epoch 200/200
116/116 [==============================] - 0s 679us/step - loss: 1.1394 - accuracy: 0.4784
```

In [18]:

```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*100:.2f}%, Accuracy: {model_accuracy*100:.2f}%")
```

```
39/39 - 0s - loss: 1.2470 - accuracy: 0.4219 - 91ms/epoch - 2ms/step
Loss: 124.70%, Accuracy: 42.19%
```

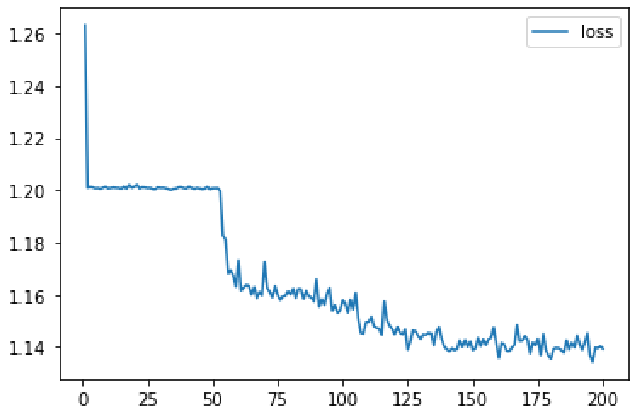
In [19]:

```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[19]:

```
<AxesSubplot:>
```

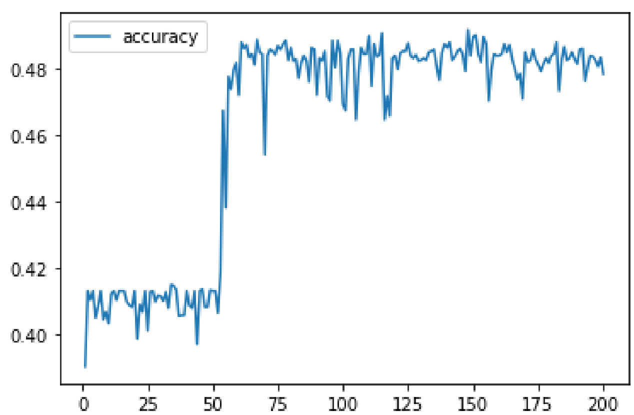


In [20]:

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

Out[20]:

```
<AxesSubplot:>
```



In [ ]: