```
import spacy
print(spacy.__version__)
```

```
3.8.5
```

```
import json
from spacy.tokens import DocBin
from sklearn.model_selection import train_test_split
```

```
from google.colab import files
uploaded = files.upload()  # Upload `real_estate_data_v3.csv`
```

Choose Files real_estate_data_v3.csv

real_estate_data_v3.csv(text/csv) - 479543 bytes, last modified: n/a - 100% done

```
Saving real_estate_data_v3.csv to real_estate_data_v3.csv
```

```
import pandas as pd
df = pd.read_csv("real_estate_data_v3.csv")
```

```
# Clean price column and generate description
df['Price'] = df['Price'].astype(str).str.replace(",", "").astype(float)

df['description'] = df.apply(lambda row:
    f"This {row['Bedrooms']}-bedroom, {row['Bathrooms']}-bathroom "
    f"{row['Property Type']} is located in {row['City']}, CA. "
    f"It has {row['Square Footage']} sqft of space and is priced at "
    f"${int(row['Price'])}. It is currently listed as {row['listing_type'].lower(
    f"Includes kitchen, living room and dining area.",
    axis=1)
```

```
def create_examples(df, n=60):
    examples = []
    for desc in df['description'].sample(n, random_state=42):
        entities = []
        for word, label in [
            ("bedroom", "BEDROOM"), ("bathroom", "BATHROOM"),
            ("apartment", "PROPERTY_TYPE"), ("single family", "PROPERTY_TYPE"),
            ("San Jose", "CITY"), ("rent", "LISTING_TYPE"), ("sale", "LISTING_TYPE'
            ("kitchen", "ROOM_TYPE"), ("living room", "ROOM_TYPE"), ("dining", "ROC
        ]:
            if word in desc:
                start = desc.index(word)
                end = start + len(word)
                entities.append((start, end, label))
        examples.append((desc, {"entities": entities}))
    return examples

all_data = create_examples(df)
train_data, dev_data = train_test_split(all_data, test_size=0.2, random_state=42)
```

```
from google.colab import files
uploaded = files.upload()
```

Choose Files train_data_final_full.json

train_data_final_full.json(application/json) - 40645 bytes, last modified: n/a - 100% done

```
Saving train_data_final_full.json to train_data_final_full.json
```

```
from sklearn.model_selection import train_test_split

# Load merged training data
with open("train_data_final_full.json", "r") as f:
    data = json.load(f)

# Split into training and dev sets
train_data, dev_data = train_test_split(data, test_size=0.15, random_state=42)
```

```
nlp = spacy.blank("en")  # fresh pipeline

def convert_to_docbin(data, label="train"):
    doc_bin = DocBin()
    for text, annots in data:
        doc = nlp.make_doc(text)
        ents = []
```

```
        for start, end, label in annots["entities"]:
            span = doc.char_span(start, end, label, alignment_mode="contract")
            if span:
                ents.append(span)
            else:
                print(f"⚠️ Dropped span: '{text[start:end]}' as {label}")
        doc.ents = ents
        doc_bin.add(doc)
    print(f"✅ Converted {len(doc_bin)} examples to {label}.spacy")
    return doc_bin

convert_to_docbin(train_data, "train").to_disk("train.spacy")
convert_to_docbin(dev_data, "dev").to_disk("dev.spacy")
```

```
⚠️ Dropped span: 'operties w' as PROPERTY_TYPE
⚠️ Dropped span: 'rage a' as ROOM_TYPE
⚠️ Dropped span: 'tchen s' as ROOM_TYPE
⚠️ Dropped span: 'r vill' as PROPERTY_TYPE
⚠️ Dropped span: 'r sa' as LISTING_TYPE
⚠️ Dropped span: 'ttached ba' as BATHROOM
⚠️ Dropped span: 'ouses ' as PROPERTY_TYPE
⚠️ Dropped span: ' Jose.' as CITY
⚠️ Dropped span: 'illa ' as PROPERTY_TYPE
⚠️ Dropped span: 'lmaden ' as CITY
⚠️ Dropped span: 'ckyard.' as ROOM_TYPE
⚠️ Dropped span: 'bathroom' as ROOM_TYPE
⚠️ Dropped span: 'bathroom' as ROOM_TYPE
⚠️ Dropped span: 'plex h' as PROPERTY_TYPE
⚠️ Dropped span: 'mes w' as PROPERTY_TYPE
⚠️ Dropped span: 'bathroom' as ROOM_TYPE
⚠️ Dropped span: 'bathroom' as ROOM_TYPE
⚠️ Dropped span: 'HK c' as PROPERTY_TYPE
⚠️ Dropped span: 'ndos f' as PROPERTY_TYPE
⚠️ Dropped span: 'ent ' as LISTING_TYPE
⚠️ Dropped span: 'n Jose?' as CITY
⚠️ Dropped span: 'uses w' as PROPERTY_TYPE
⚠️ Dropped span: 'uses w' as PROPERTY_TYPE
⚠️ Dropped span: 'ths.' as BATHROOM
⚠️ Dropped span: 'bathroom' as ROOM_TYPE
⚠️ Dropped span: ' condo' as PROPERTY_TYPE
⚠️ Dropped span: 'partment ' as PROPERTY_TYPE
⚠️ Dropped span: 'nt i' as LISTING_TYPE
⚠️ Dropped span: 'n Jose.' as CITY
⚠️ Dropped span: 'bathroom' as ROOM_TYPE
⚠️ Dropped span: 'partment ' as PROPERTY_TYPE
⚠️ Dropped span: 'bathroom' as ROOM_TYPE
⚠️ Dropped span: 'ouses ' as PROPERTY_TYPE
⚠️ Dropped span: 'aden ar' as CITY
```

```
⚠️ Dropped span: 'artment i' as PROPERTY_TYPE
⚠️ Dropped span: ' Cambria' as CITY
⚠️ Dropped span: 'd balco' as ROOM_TYPE
⚠️ Dropped span: 'illa ' as PROPERTY_TYPE
⚠️ Dropped span: 'arden ' as ROOM_TYPE
⚠️ Dropped span: 'partments ' as PROPERTY_TYPE
⚠️ Dropped span: 'alcony ' as ROOM_TYPE
⚠️ Dropped span: 'bathroom' as ROOM_TYPE
⚠️ Dropped span: 'bathroom' as ROOM_TYPE
⚠️ Dropped span: 'bathroom' as ROOM_TYPE
⚠️ Dropped span: 'bathroom' as ROOM_TYPE
⚠️ Dropped span: 'partments ' as PROPERTY_TYPE
⚠️ Dropped span: 'ondo ' as PROPERTY_TYPE
⚠️ Dropped span: 'ous kit' as ROOM_TYPE
✅ Converted 73 examples to ROOM_TYPE.spacy
⚠️ Dropped span: 'bathroom' as ROOM_TYPE
⚠️ Dropped span: 'bathroom' as ROOM_TYPE
⚠️ Dropped span: 'bathroom' as ROOM_TYPE
⚠️ Dropped span: 'omes ' as PROPERTY_TYPE
⚠️ Dropped span: 'itchen ' as ROOM_TYPE
⚠️ Dropped span: 'ownhouses i' as PROPERTY_TYPE
⚠️ Dropped span: 'n Jose.' as CITY
⚠️ Dropped span: ' garag' as ROOM_TYPE
✅ Converted 14 examples to CITY.spacy
```

```
!python -m spacy init config config.cfg --lang en --pipeline ner --force
```

```
⚠ To generate a more effective transformer-based config (GPU-only),
install the spacy-transformers package and re-run this command. The config
generated now does not use transformers.
ℹ Generated config template specific for your use case
- Language: en
- Pipeline: ner
- Optimize for: efficiency
- Hardware: CPU
- Transformer: None
✔ Auto-filled config with all values
✔ Saved config
config.cfg
You can now add your data and train your pipeline:
python -m spacy train config.cfg --paths.train ./train.spacy --paths.dev ./dev
```

```
!python -m spacy train config.cfg --output ./output \
  --paths.train ./train.spacy \
  --paths.dev ./dev.spacy \
  --gpu-id -1
```

```
ℹ Saving to output directory: output
ℹ Using CPU

========================== Initializing pipeline ==========================
✔ Initialized pipeline

============================= Training pipeline =============================
ℹ Pipeline: ['tok2vec', 'ner']
ℹ Initial learn rate: 0.001
E    #       LOSS TOK2VEC  LOSS NER  ENTS_F  ENTS_P  ENTS_R  SCORE
---  ------  ------------  --------  ------  ------  ------  ------
  0       0          0.00     62.71    0.00    0.00    0.00    0.00
 16     200        131.72   2813.37   80.73   86.27   75.86    0.81
 37     400         84.24    160.02   80.00   84.62   75.86    0.80
 63     600         74.83     81.01   78.95   80.36   77.59    0.79
 94     800         75.50     31.95   78.57   81.48   75.86    0.79
133    1000         52.29     14.05   80.73   86.27   75.86    0.81
180    1200        102.70     23.89   80.00   80.70   79.31    0.80
236    1400         46.31      7.53   78.57   81.48   75.86    0.79
303    1600         14.86      2.50   80.00   84.62   75.86    0.80
388    1800        124.74     18.96   82.14   85.19   79.31    0.82
488    2000        175.72     17.30   80.34   79.66   81.03    0.80
588    2200        130.26     19.49   80.36   83.33   77.59    0.80
744    2400        222.28     19.93   80.70   82.14   79.31    0.81
944    2600        361.49     23.88   79.28   83.02   75.86    0.79
1144    2800         19.56      1.93   79.63   86.00   74.14    0.80
1344    3000        207.56     22.70   79.28   83.02   75.86    0.79
1544    3200         19.40      1.82   79.28   83.02   75.86    0.79
1744    3400        199.63     14.53   80.36   83.33   77.59    0.80
✔ Saved pipeline to output directory
output/model-last
```

```
import shutil
shutil.make_archive("real_estate_nlp_model", 'zip', "output/model-best")
files.download("real_estate_nlp_model.zip")
```